Introduction to Computer Graphics
SIGGRAPH 2001

Syllabus

Course Organizer

Michael Bailey
San Diego Supercomputer Center
University of California San Diego

Course Speakers

Andrew Glassner
Independent Writer and Consultant

Computer graphics is an exciting field of endeavor, but it is often difficult for a newcomer to get started. This course is that opportunity. The topics being presented will address many areas within computer graphics and treat each from the point of view of "why-do-I-care" and "how-to." Those who take this course will emerge well-prepared to take on further study, including the taking of other SIGGRAPH courses. Attendees will also be ready to take on the vendor show and better appreciate the Electronic Theatre. We hope you enjoy reading and using these notes as much as we enjoyed preparing them.

If you have specific comments about how we can improve the course or the notes, please send them to me at: `mjb@sdsc.edu`

– Mike Bailey

Take them, use them, bring them to the masses.
Shake them, lose them, sing them to your classes.
Tiles of tides, piles of slides.
Piles of slides that no-one derides.
Slides of knowledge, slides of power – slides that last a half an hour.
Small slides. Blue slides. Old-hat and what's-new slides.
Take a slide and project it wide.
Project it far and make it tall, a slide's a slide that's seen by all.
SIGGRAPH slides go into holders, printed pages go into folders.
We teach. We teach in courses. We teach whatever the market enforces.
You want pixels? You want rays?
We'll lead you through the graphics maze.

– Andrew Glassner

SIGGRAPH ‘01
Introduction to Computer Graphics
About the Speakers

Michael J. Bailey

Mike Bailey is a researcher at the San Diego Supercomputer Center and an adjunct professor in the Computer Science and Mechanical Engineering departments at the University of California at San Diego. Mike received his Ph.D. from Purdue University. He has also worked at Sandia National Laboratories, Purdue University, Megatek, SDSC, and UCSD. Mike’s areas of interest include scientific visualization, computer aided design, and solid freeform fabrication. He has authored numerous papers on the use of computer graphics in engineering and science. Mike founded the interdisciplinary Design Visualization Lab at SDSC/UCSD, which includes the Center for Visualization Prototypes which applies solid freeform fabrication methods to visualization problems. Mike has served on the SIGGRAPH Executive Committee and was SIGGRAPH conference co-chair in 1991. Mike has also served as SIGGRAPH Courses Chair in 1984, 1985, 1987, 1988, and 1994.

Andrew S. Glassner

Dr. Andrew Glassner is an independent writer and consultant. He has worked at the NYIT Computer Graphics Lab, Case Western Reserve University, the IBM TJ Watson Research Lab, the Delft University of Technology, Bell Communications Research, Xerox PARC, and Microsoft Research. He has published numerous technical papers on topics ranging from digital sound to new rendering techniques. His book *3D Computer Graphics: A Handbook for Artists and Designers* has taught a generation of artists. Glassner created and edited the *Graphics Gems* book series and the book *An Introduction to Ray Tracing*. His most recent text is *Principles of Digital Image Synthesis*, a two-volume treatise on rendering theory and practice published by Morgan-Kaufmann. Andrew served SIGGRAPH '94 as Chair of the Papers Committee, and creator of the Sketches venue. He has also served as Founding Editor of the *Journal of Graphics Tools*, and Editor-in-Chief of *ACM Transactions on Graphics*. In his free time Andrew plays jazz piano, draws, and writes fiction. He holds a Ph.D. in Computer Science from the University of North Carolina at Chapel Hill.

SIGGRAPH 2001
Introduction to Computer Graphics

Mike Bailey (M)
Andrew Glassner (A)

Course Schedule

8:30 - 9:00	Welcome Overview of the Course Overview of the Graphics Process Some graphics to look at	M
9:00 – 10:00	Modeling	A
10:00 – 10:15	**Morning Break**	
10:15 – 11:15	Rendering	A
11:15 – 12:00	Graphics display hardware	M
12:00 – 1:30	**Lunch**	
1:30 – 2:30	Animation	A
2:30 – 3:00	Scientific Visualization	M
3:00 – 3:15	**Afternoon Break**	
3:15 – 3:30	More Scientific Visualization	M
3:30 – 3:45	Input devices	M
3:45 – 4:00	Graphics on the World Wide Web	M
4:00 – 4:15	How to attend a SIGGRAPH	M
4:15 – 4:30	Finding additional information	M
4:30 – 5:00	Questions and general discussion	A & M

SIGGRAPH 2001
Introduction to Computer Graphics

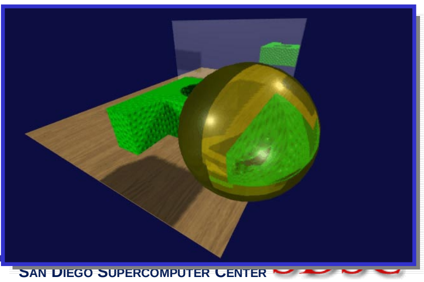
Course Note Table of Contents

Introduction to Computer Graphics

SIGGRAPH 2001

Mike Bailey — San Diego Supercomputer Center and University of California San Diego

Andrew Glassner — Independent writer/consultant



Mike Bailey

PhD from Purdue University

Has worked at Sandia Labs, Purdue University, Megatek, San Diego Supercomputer Center, and the University of California at San Diego

`mjb@sdsc.edu`

Andrew Glassner

PhD from the University of North Carolina - Chapel Hill

Has worked at IBM, Bell Communications, Delft University, NYIT, Xerox PARC, and Microsoft Research

`andrew_glassner@yahoo.com`

Course Goals

- Provide a background for papers, panels, and other courses
- Help appreciate the Electronic Theater
- Get more from the vendor exhibits
- Give our take on where the future is
- Provide pointers for further study

Topics

- Overview of the “Graphics Process” (Mike)
- Modeling (Andrew)
- Rendering (Andrew)
- Display Hardware (Mike)

More Topics

- Animation (Andrew)
- Scientific Visualization (Mike)
- Input Devices (Mike)

And, Even More Topics !

- Graphics on the Web (Mike)
- How to Attend a SIGGRAPH (Mike)
- Finding Additional Information (Mike)
- Questions and General Discussion (Andrew & Mike)

The Graphics Process

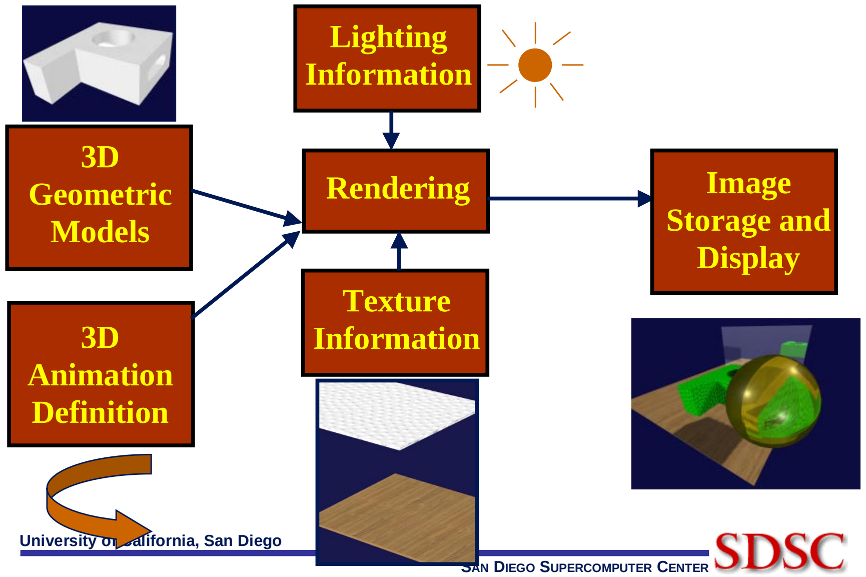
Mike Bailey

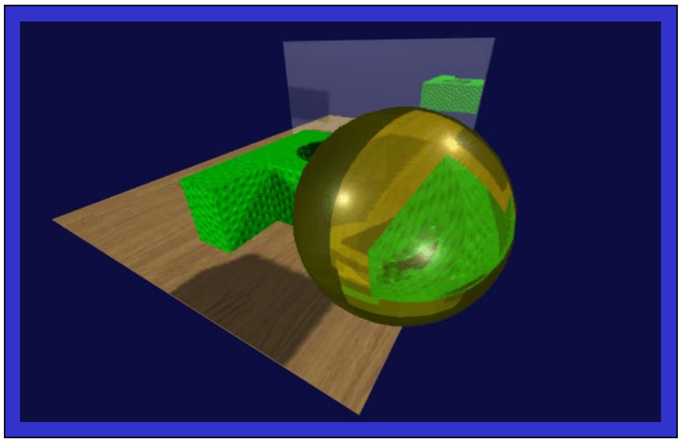
San Diego Supercomputer Center
University of California San Diego

`mjb@sdsc.edu`

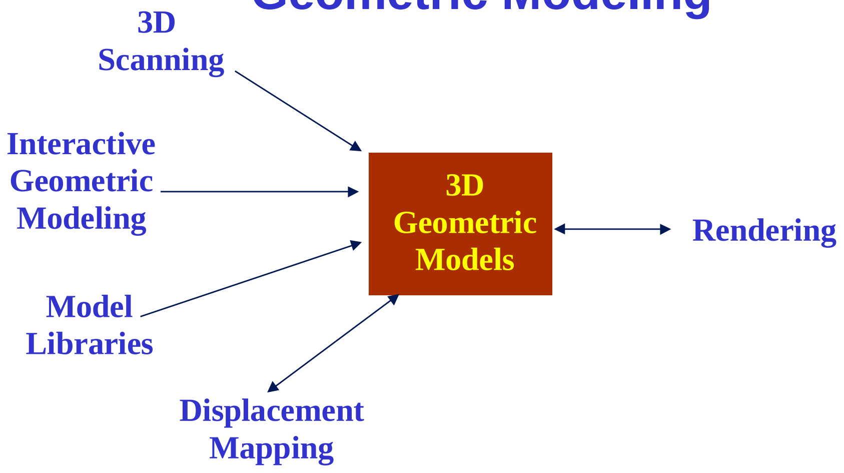
`http://www.sdsc.edu/~mjb`

The Graphics Process

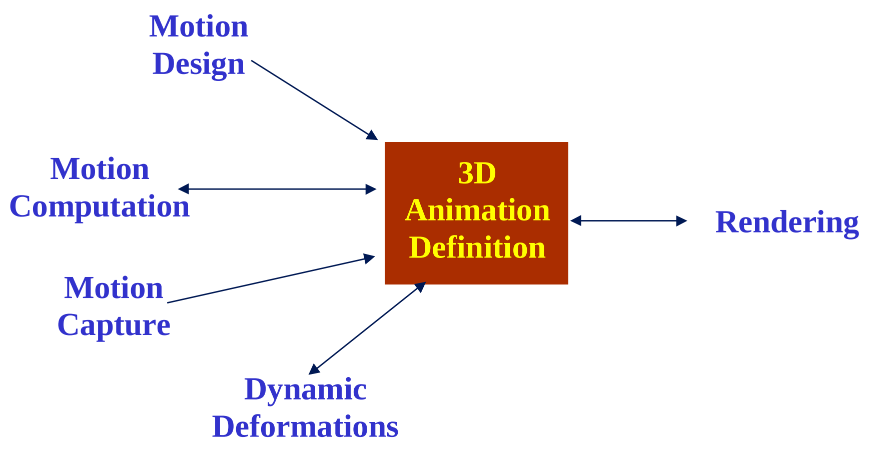




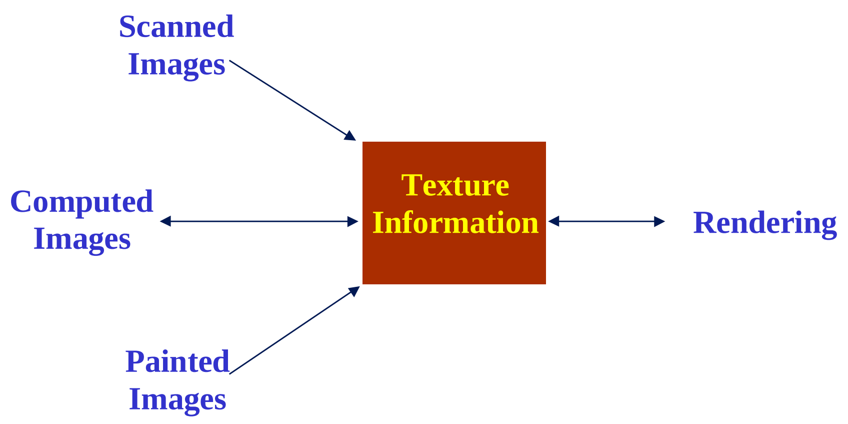
The Graphics Process: Geometric Modeling



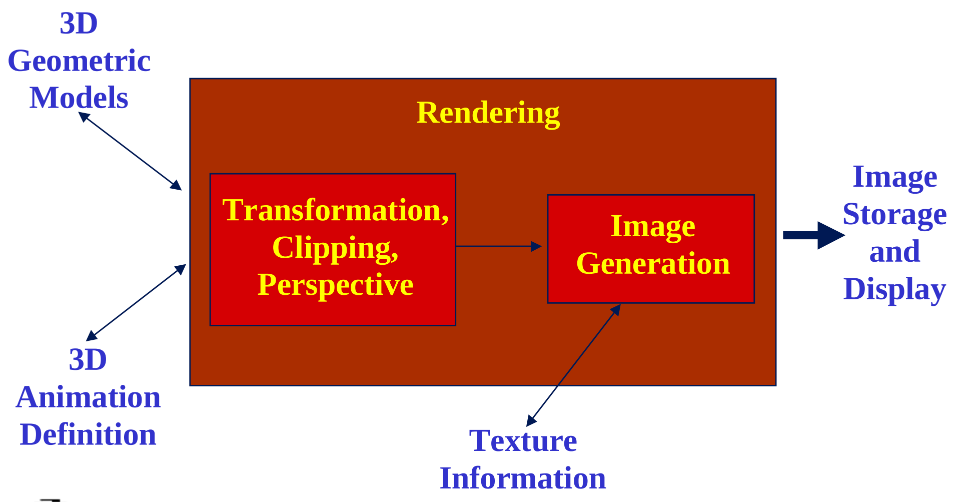
The Graphics Process: 3D Animation



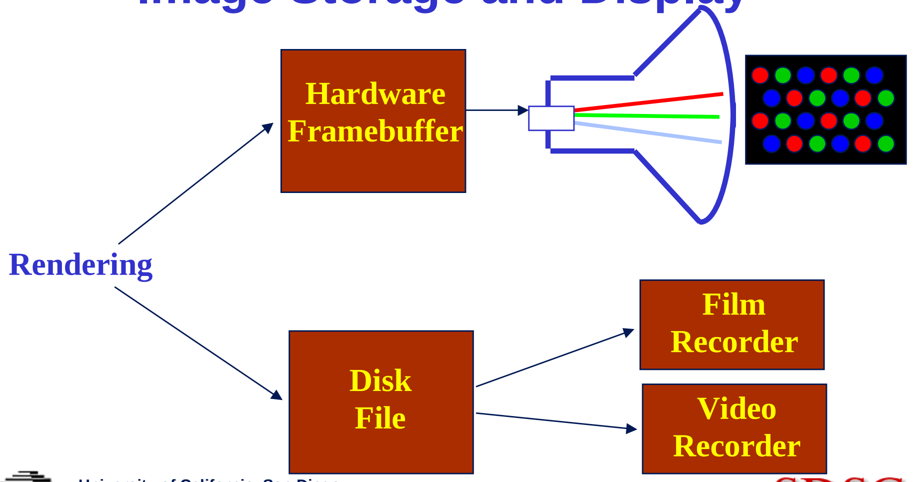
The Graphics Process: Texturing

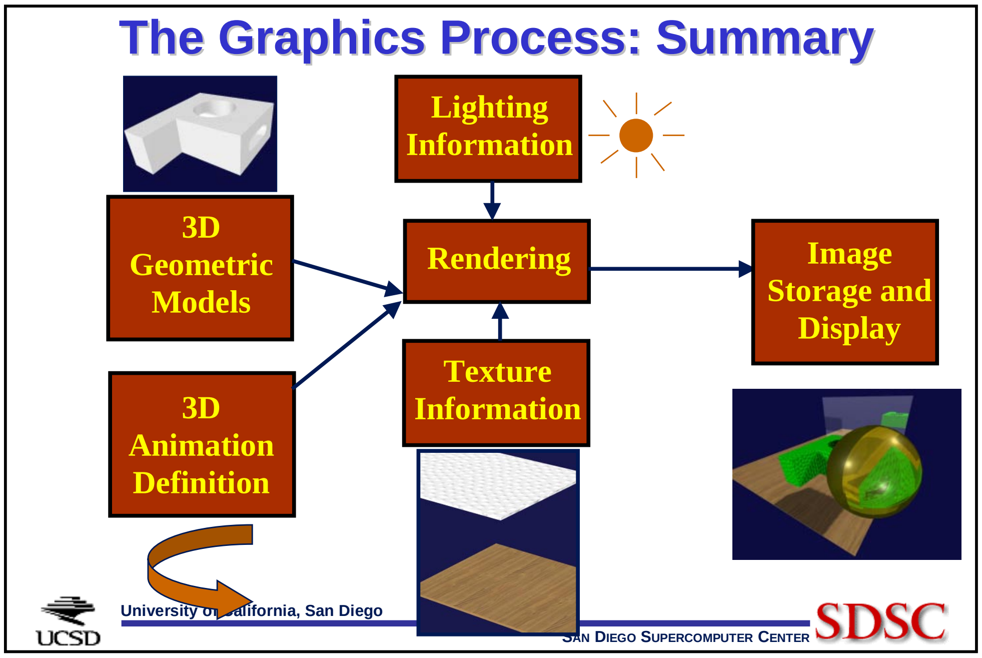


The Graphics Process: Rendering

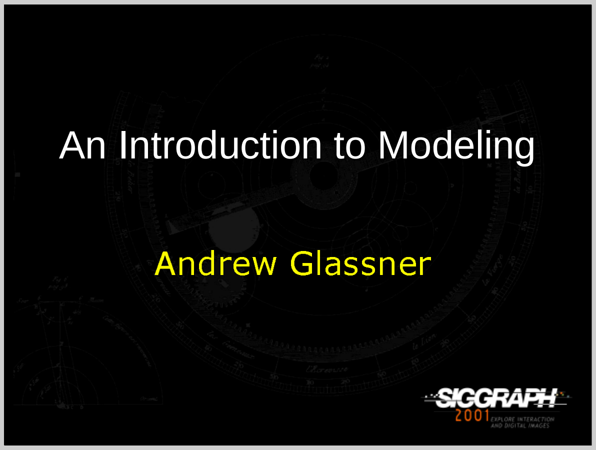


The Graphics Process: Image Storage and Display





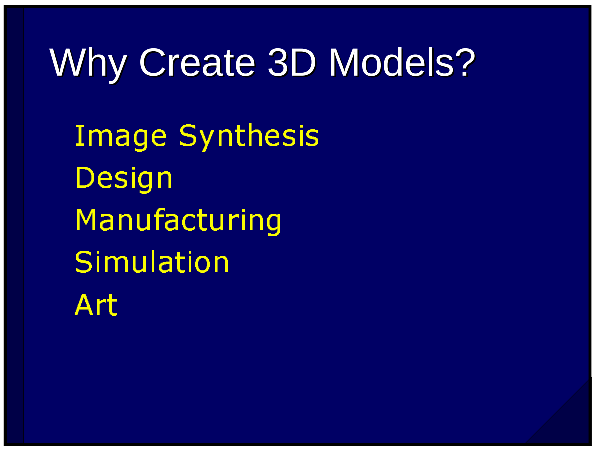
The Graphics Process: Summary



An Introduction to Modeling

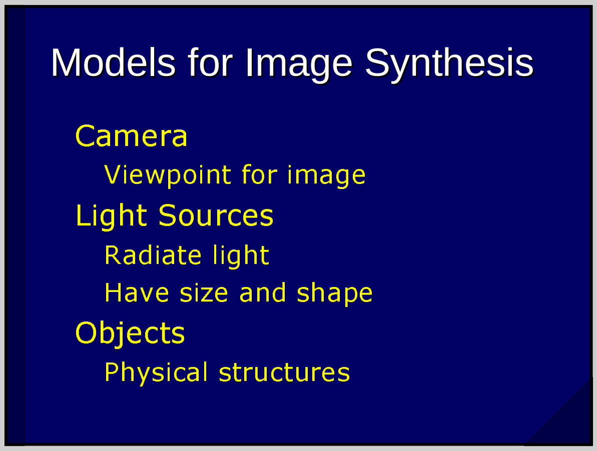
Andrew Glassner

SIGGRAPH 2001 EXPLORE INTERACTION AND DIGITAL IMAGES



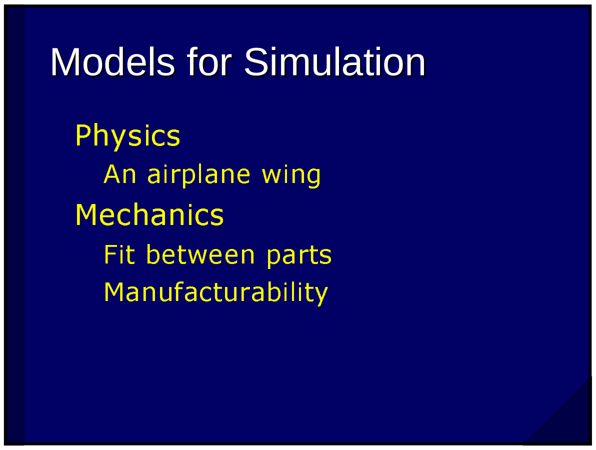
Why Create 3D Models?

- Image Synthesis
- Design
- Manufacturing
- Simulation
- Art



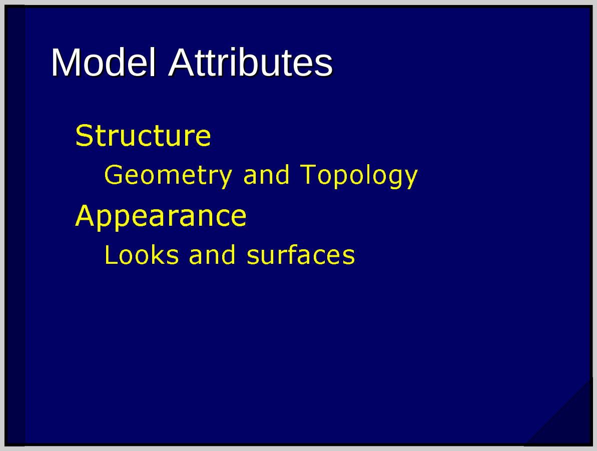
Models for Image Synthesis

- Camera
  - Viewpoint for image
- Light Sources
  - Radiate light
  - Have size and shape
- Objects
  - Physical structures



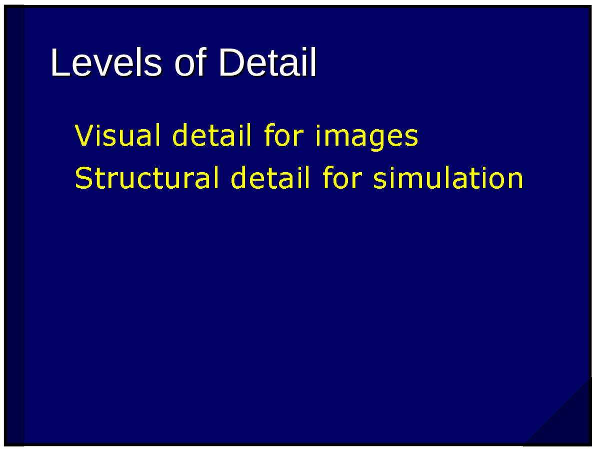
Models for Simulation

- Physics
  - An airplane wing
- Mechanics
  - Fit between parts
  - Manufacturability



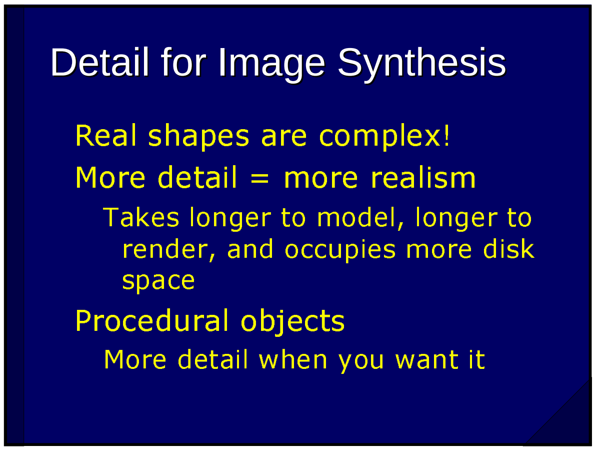
Model Attributes

- Structure
  - Geometry and Topology
- Appearance
  - Looks and surfaces



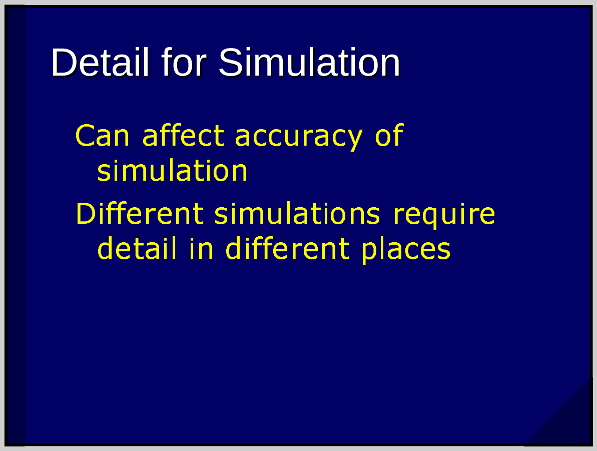
Levels of Detail

- Visual detail for images
- Structural detail for simulation



Detail for Image Synthesis

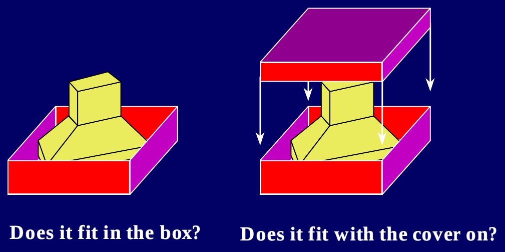
- Real shapes are complex!
- More detail = more realism
  - Takes longer to model, longer to render, and occupies more disk space
- Procedural objects
  - More detail when you want it

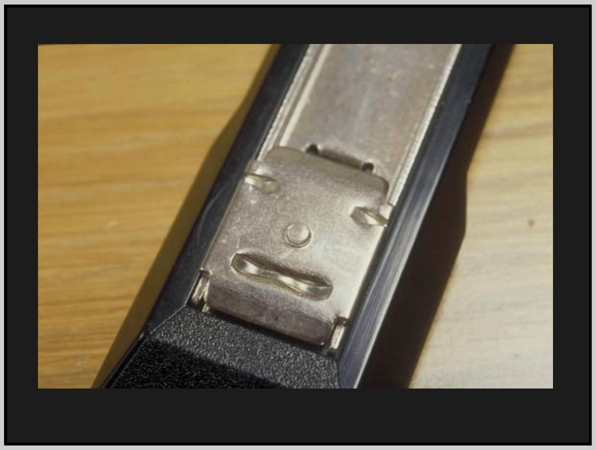


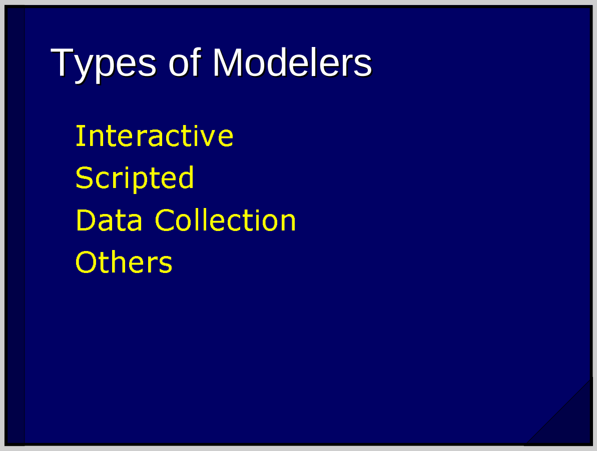
Detail for Simulation

- Can affect accuracy of simulation
- Different simulations require detail in different places

Levels of Detail for Simulations

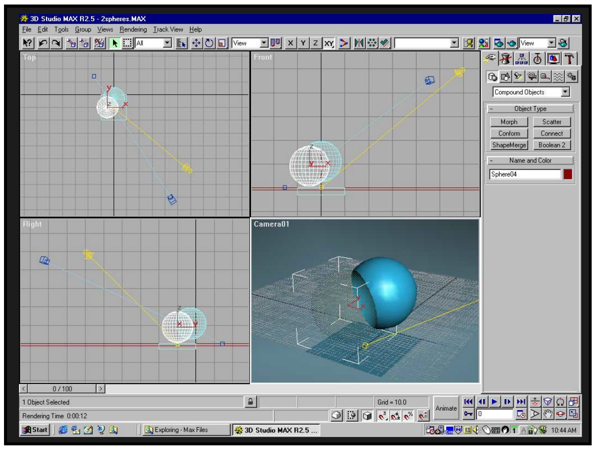


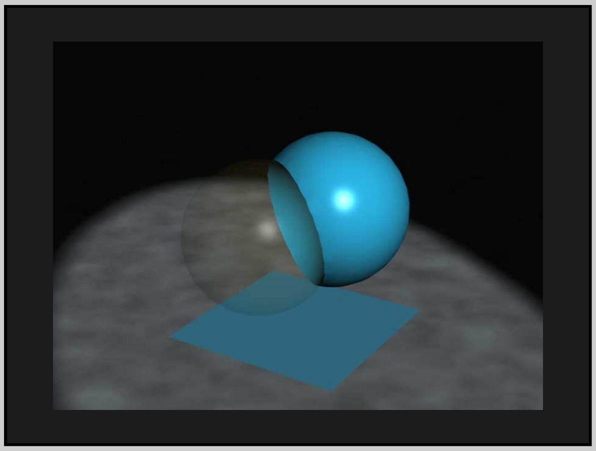


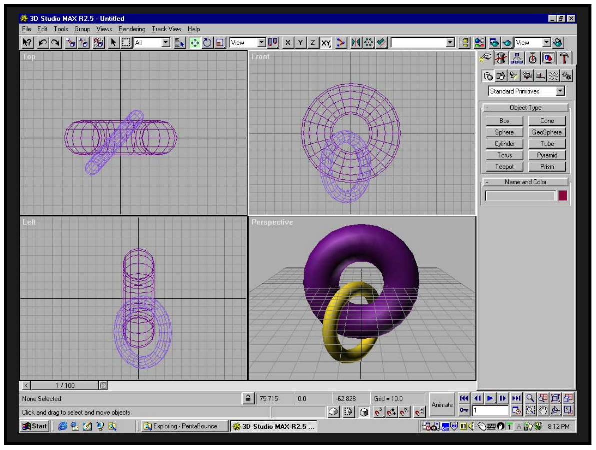


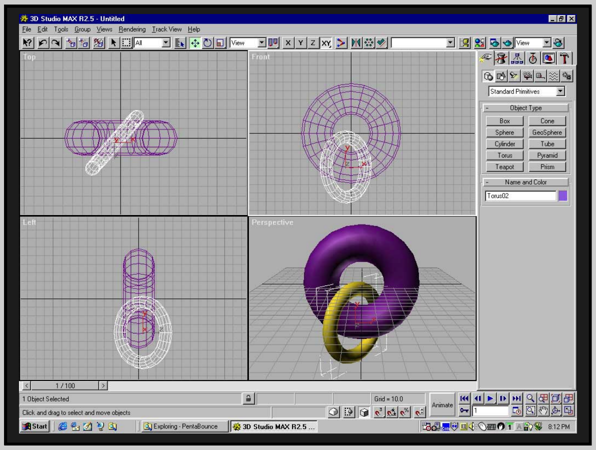
Types of Modelers

- Interactive
- Scripted
- Data Collection
- Others



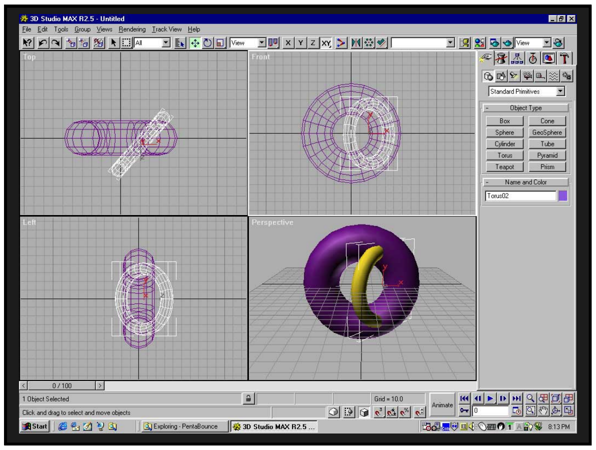






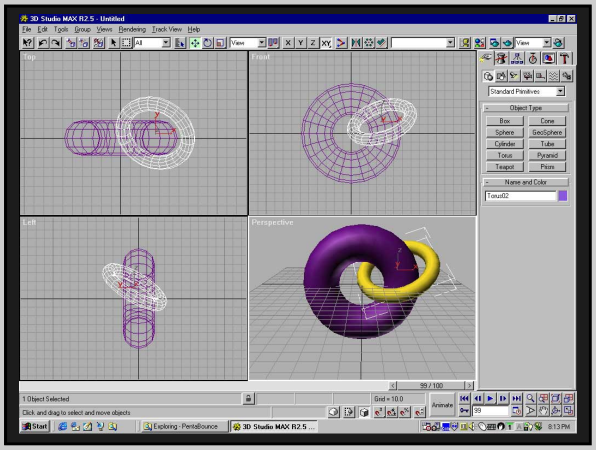
1 Object Selected

Click and drag to select and move objects



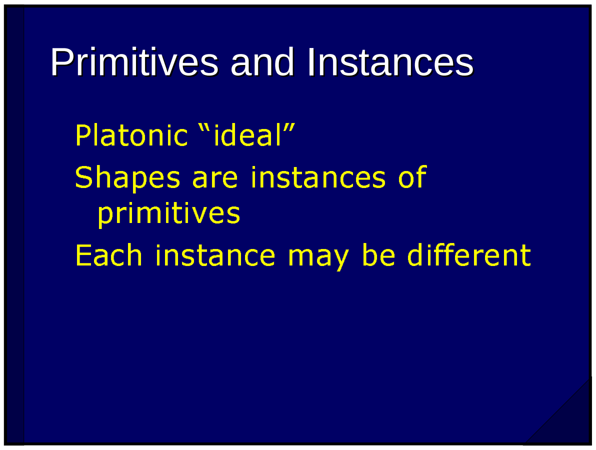
1 Object Selected

Click and drag to select and move objects



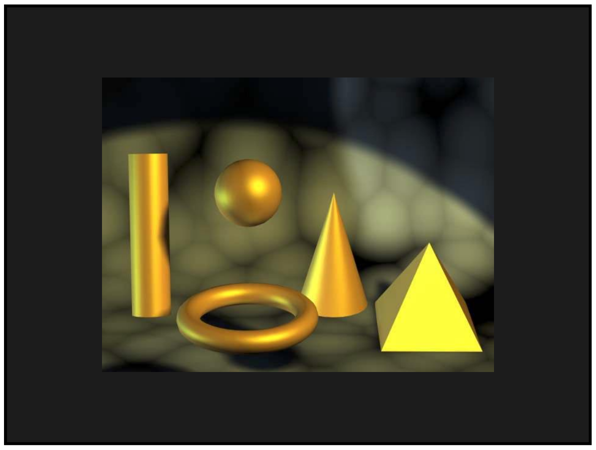
1 Object Selected

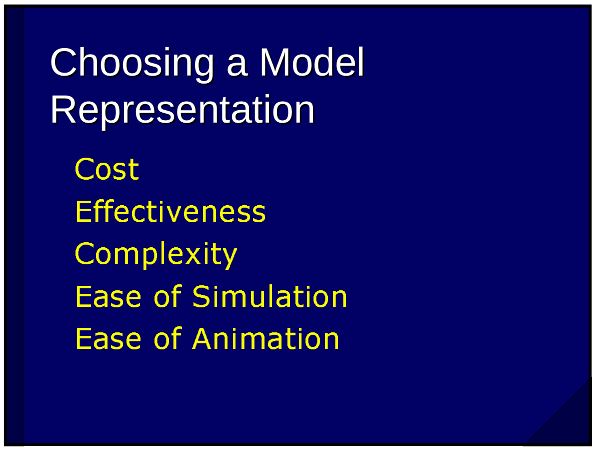
Click and drag to select and move objects



Primitives and Instances

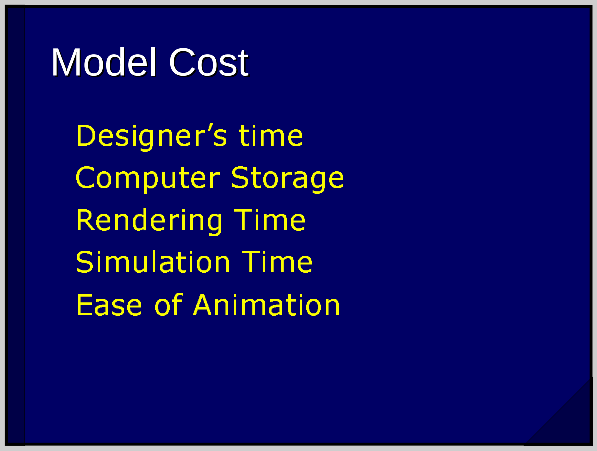
- Platonic “ideal”
- Shapes are instances of primitives
- Each instance may be different





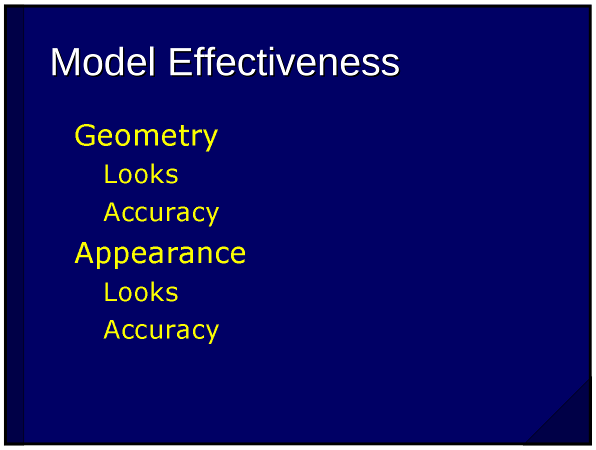
Choosing a Model Representation

- Cost
- Effectiveness
- Complexity
- Ease of Simulation
- Ease of Animation



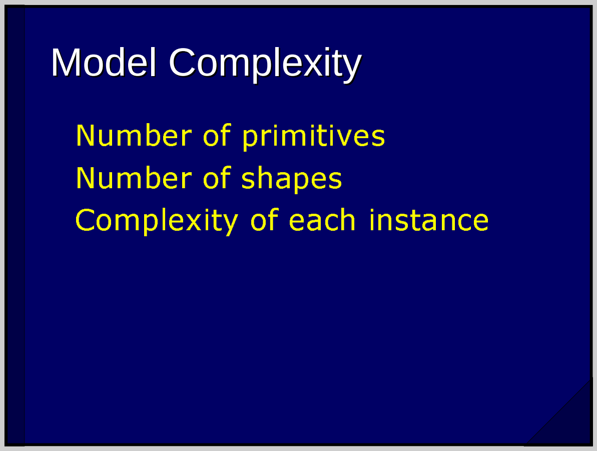
Model Cost

- Designer’s time
- Computer Storage
- Rendering Time
- Simulation Time
- Ease of Animation



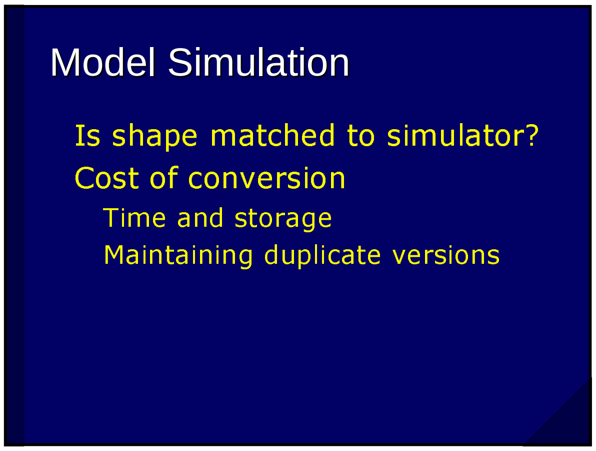
Model Effectiveness

- Geometry
  - Looks
  - Accuracy
- Appearance
  - Looks
  - Accuracy



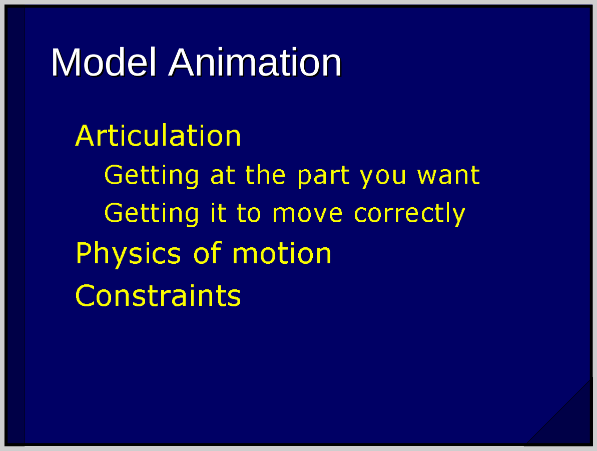
Model Complexity

- Number of primitives
- Number of shapes
- Complexity of each instance



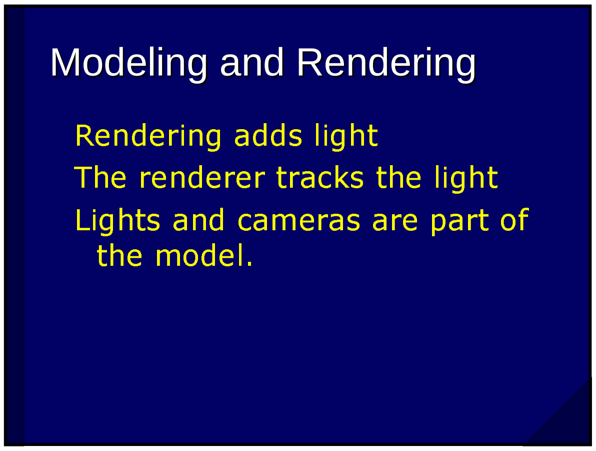
Model Simulation

- Is shape matched to simulator?
- Cost of conversion
  - Time and storage
  - Maintaining duplicate versions



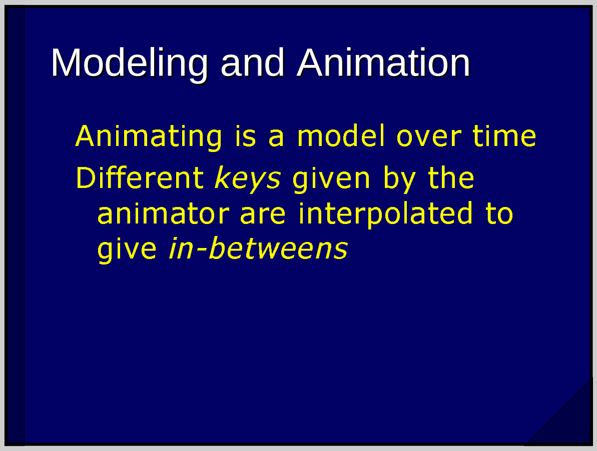
Model Animation

- Articulation
  - Getting at the part you want
  - Getting it to move correctly
- Physics of motion
- Constraints



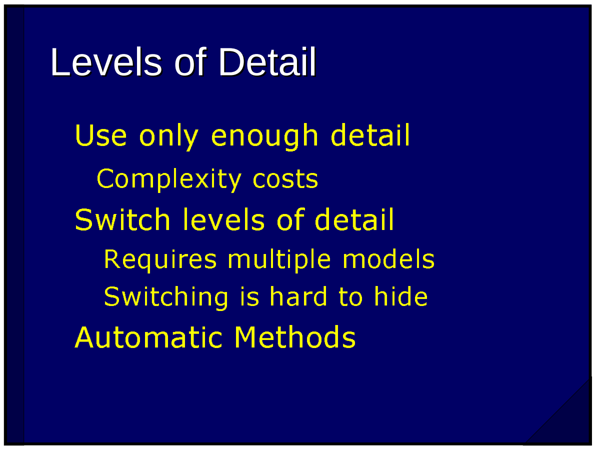
Modeling and Rendering

- Rendering adds light
- The renderer tracks the light
- Lights and cameras are part of the model.



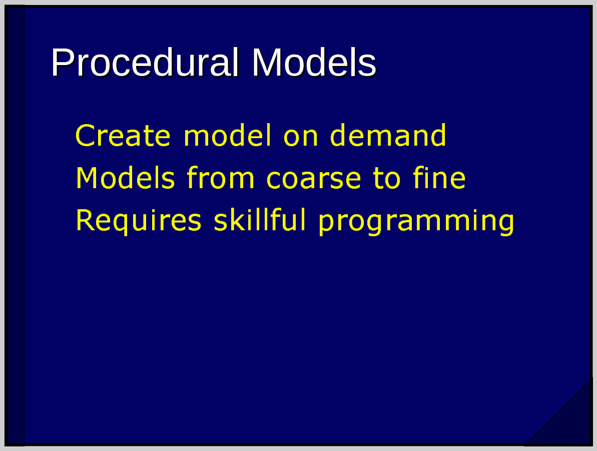
Modeling and Animation

- Animating is a model over time
- Different *keys* given by the animator are interpolated to give *in-betweens*



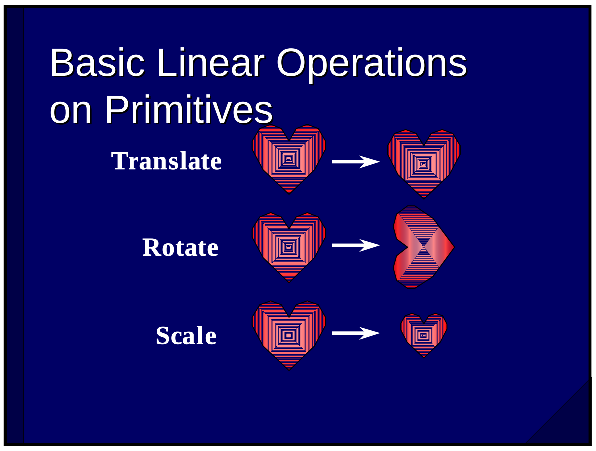
Levels of Detail

- Use only enough detail
  - Complexity costs
- Switch levels of detail
  - Requires multiple models
  - Switching is hard to hide
- Automatic Methods

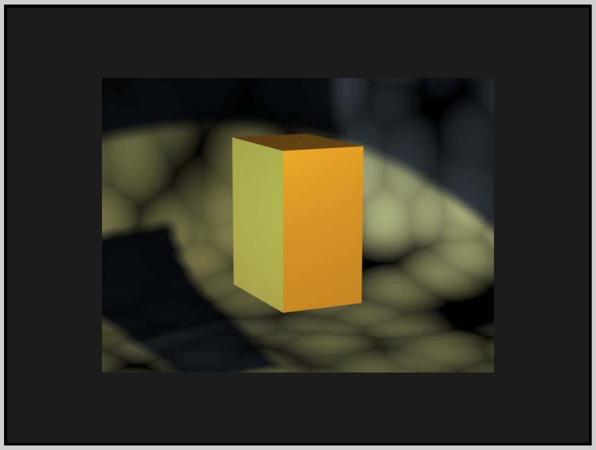


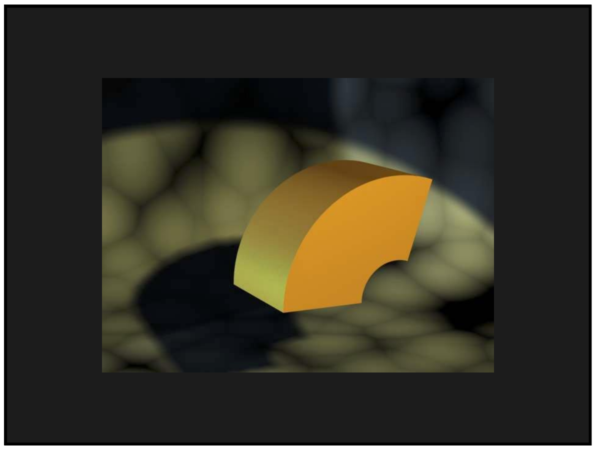
Procedural Models

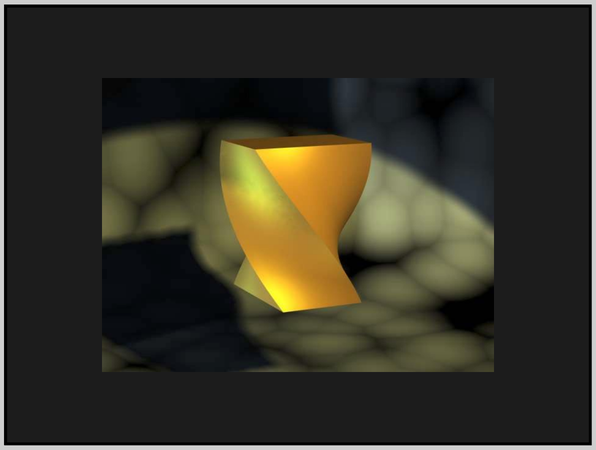
- Create model on demand
- Models from coarse to fine
- Requires skillful programming

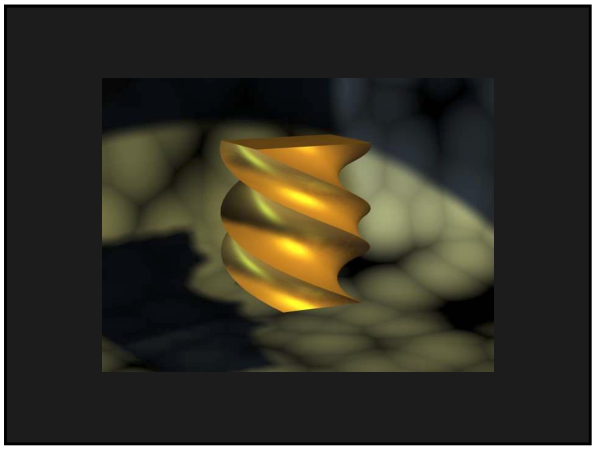


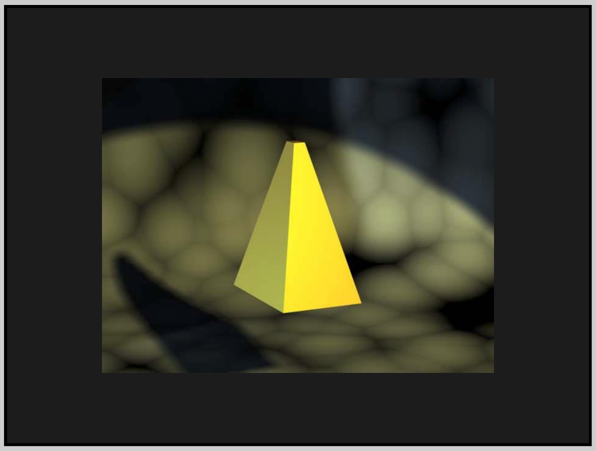
Basic Linear Operations on Primitives

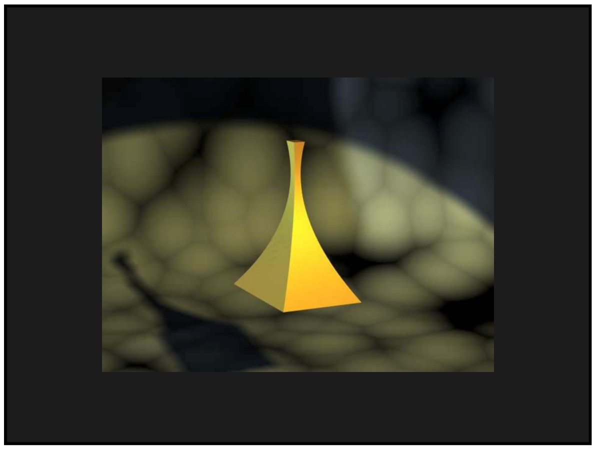


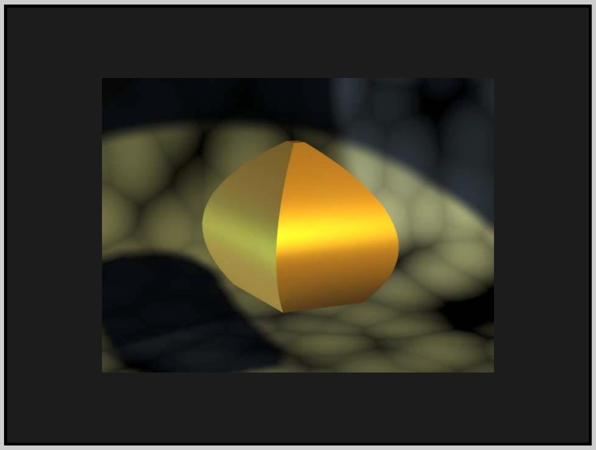
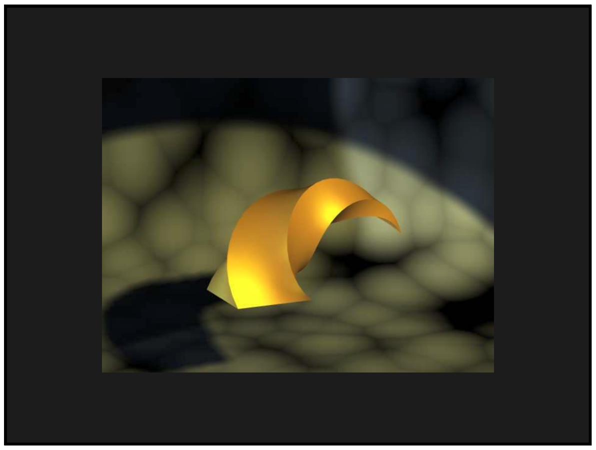
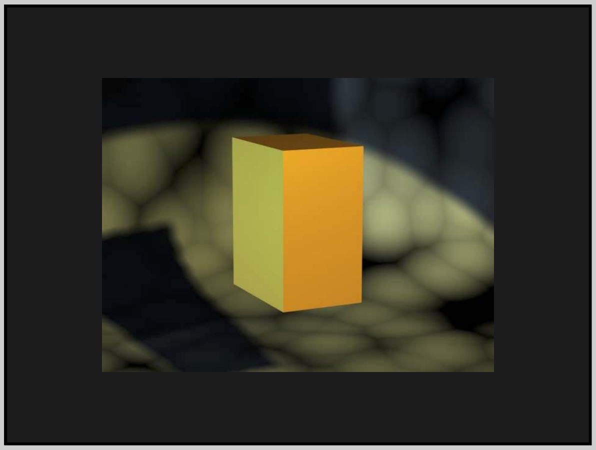
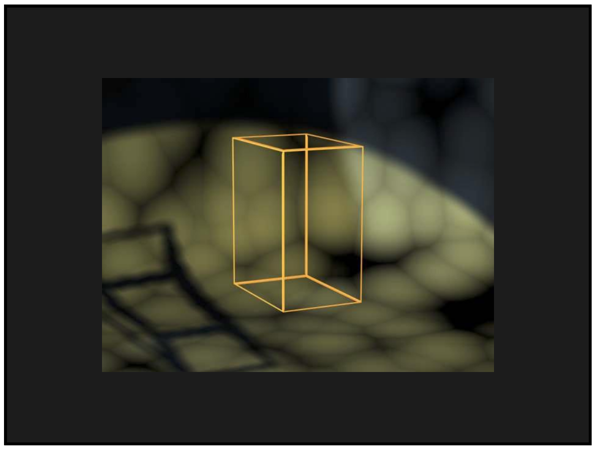
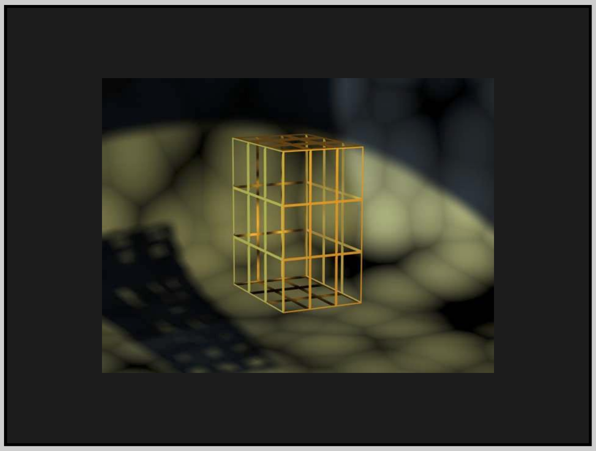
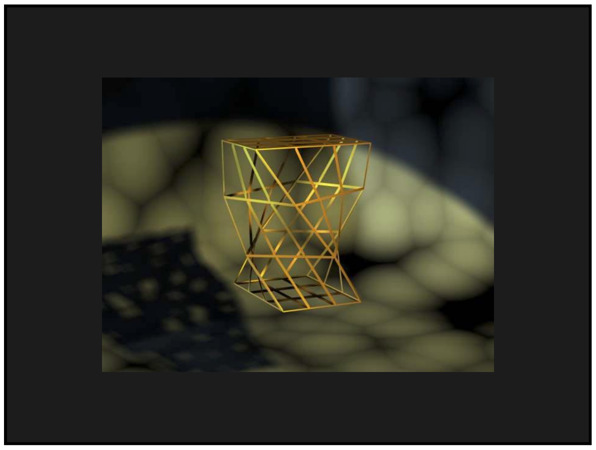


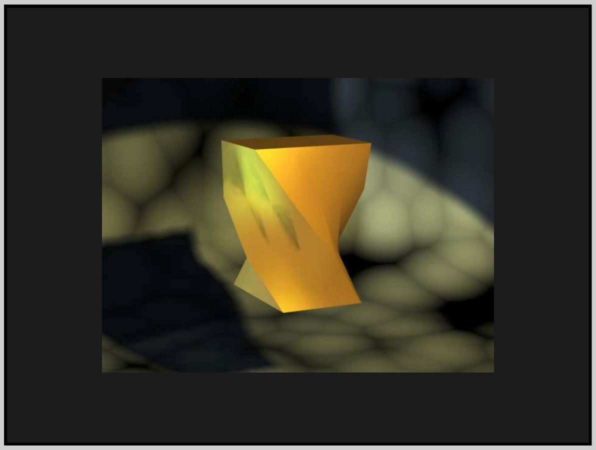


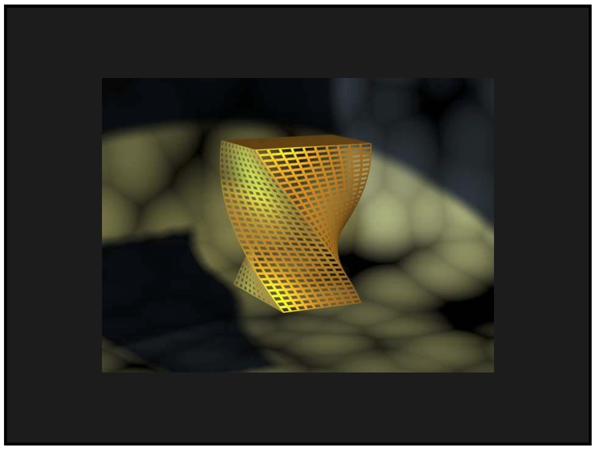


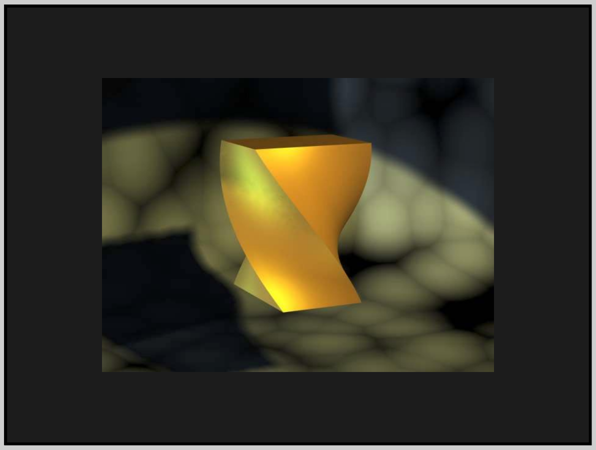


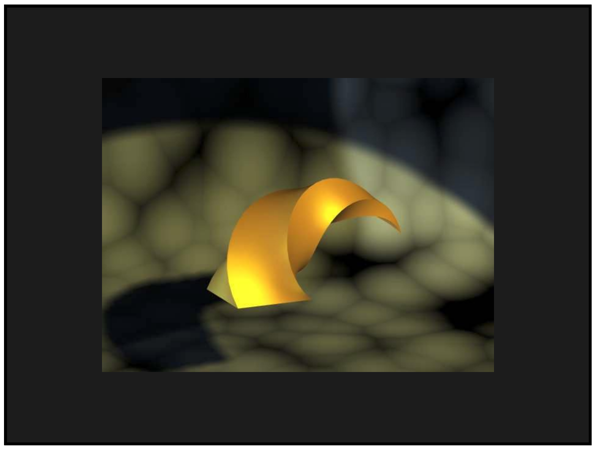


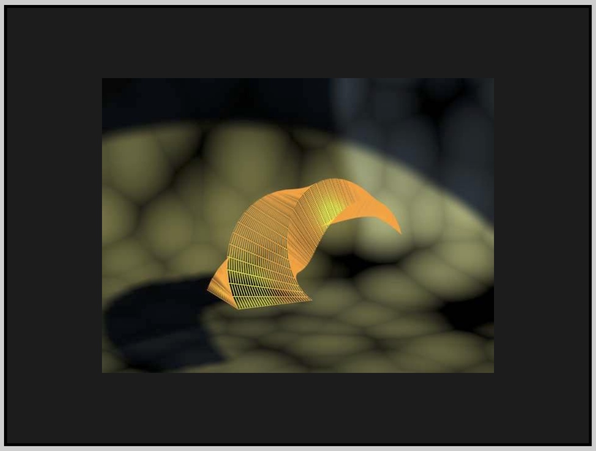







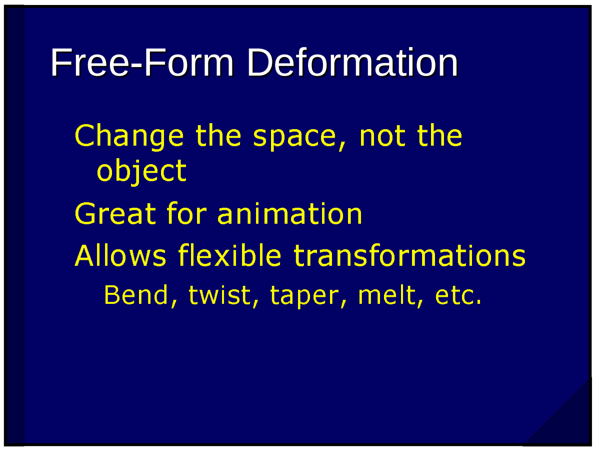






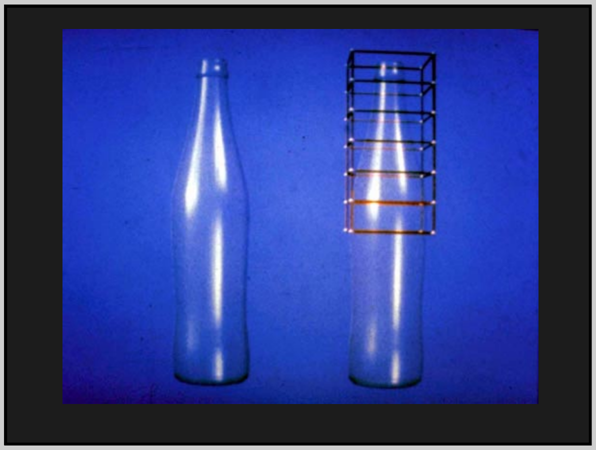


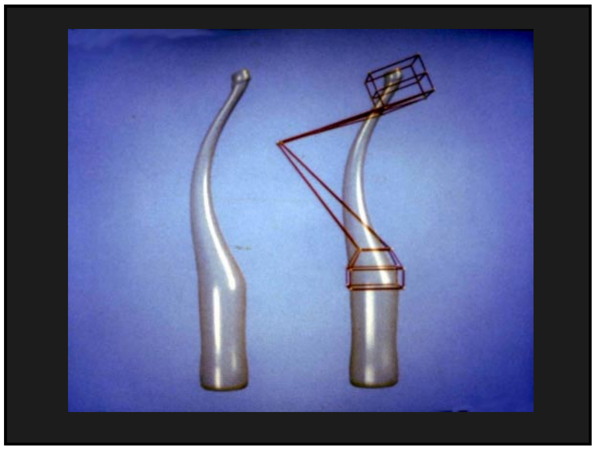


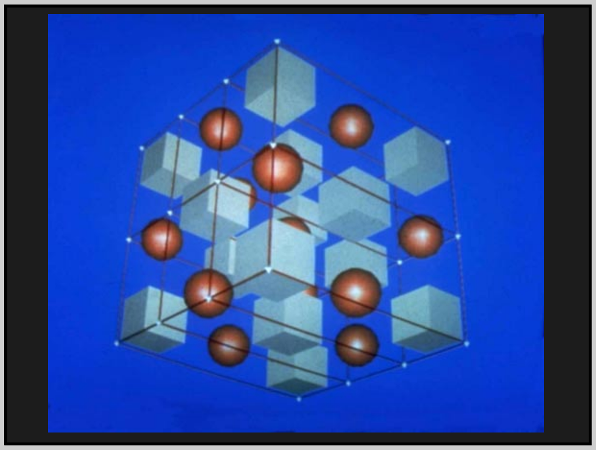


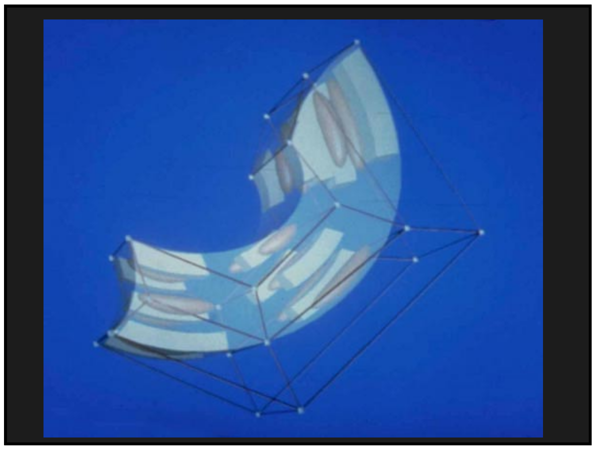
Free-Form Deformation

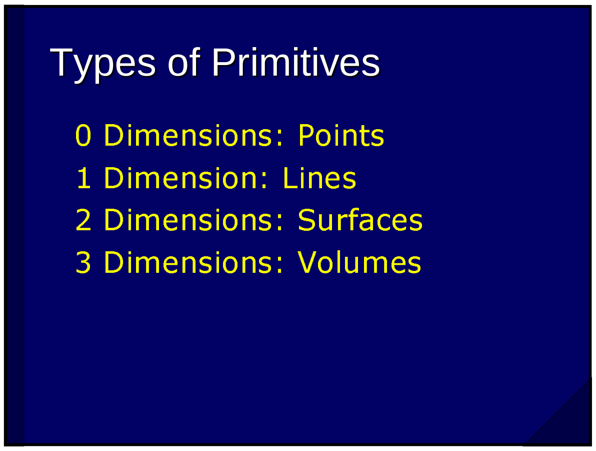
- Change the space, not the object
- Great for animation
- Allows flexible transformations
  - Bend, twist, taper, melt, etc.





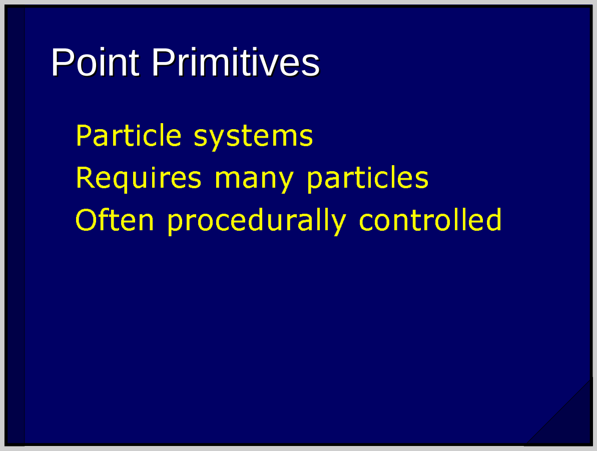






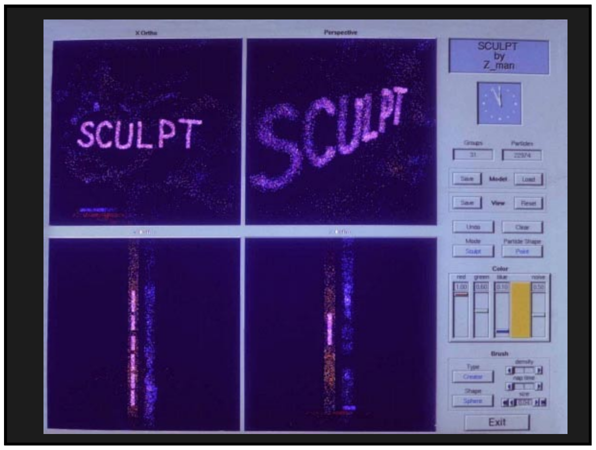
Types of Primitives

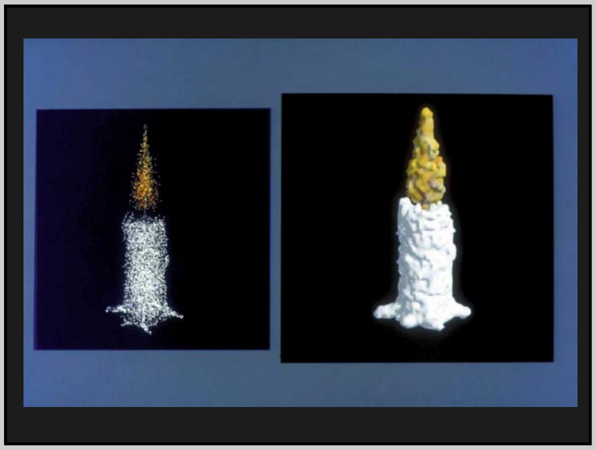
- 0 Dimensions: Points
- 1 Dimension: Lines
- 2 Dimensions: Surfaces
- 3 Dimensions: Volumes

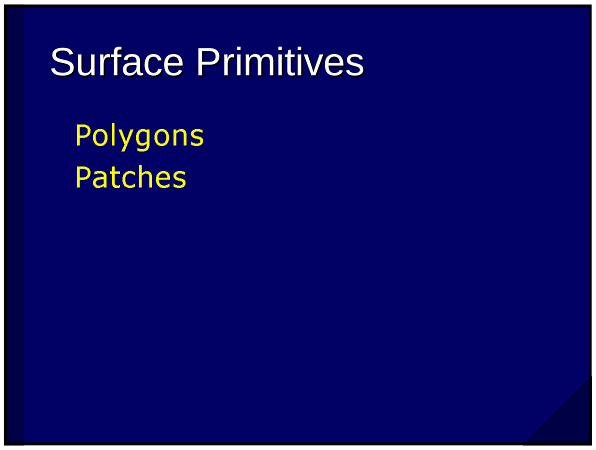


Point Primitives

- Particle systems
- Requires many particles
- Often procedurally controlled

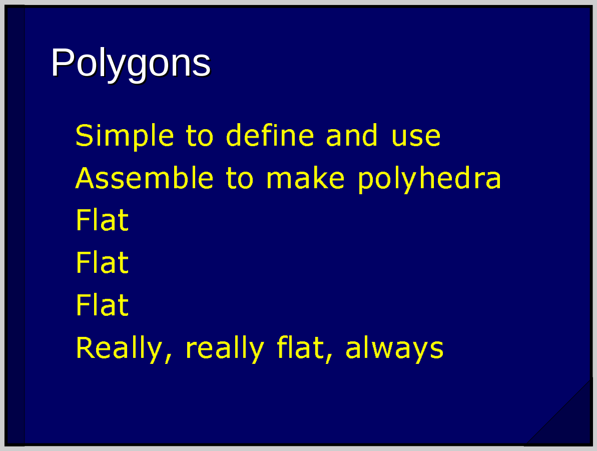






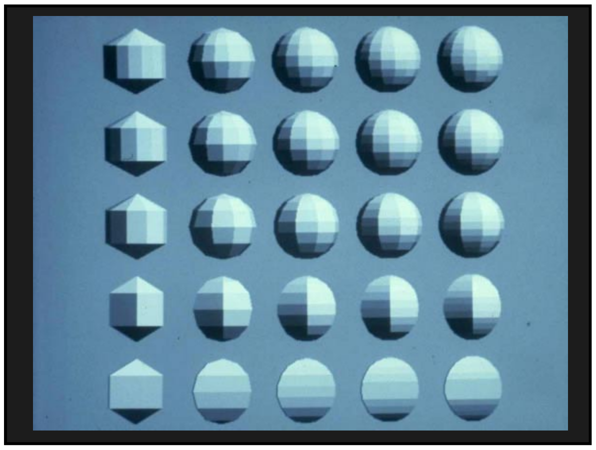
Surface Primitives

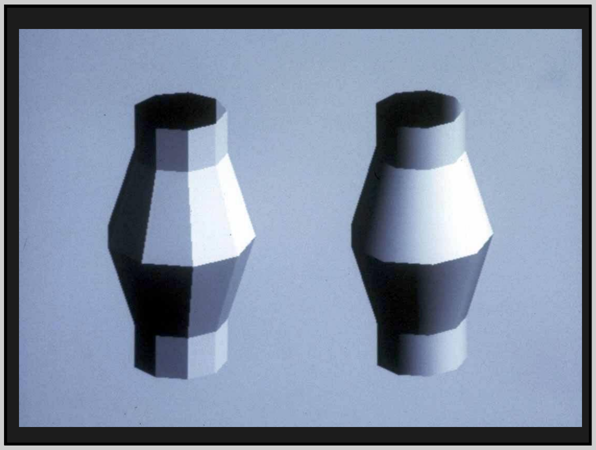
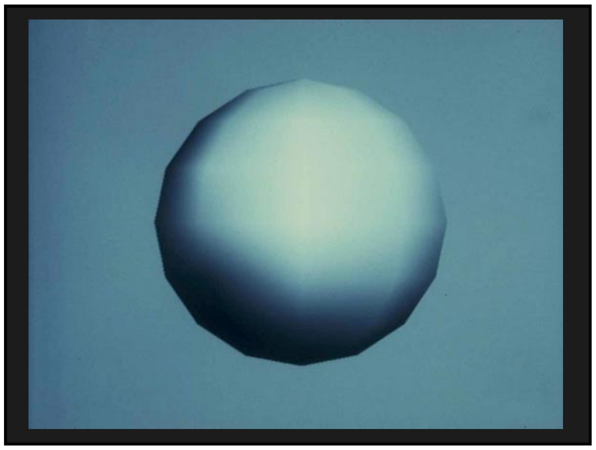
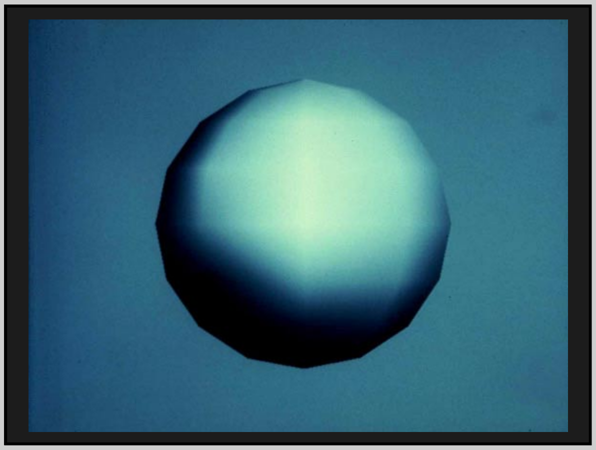
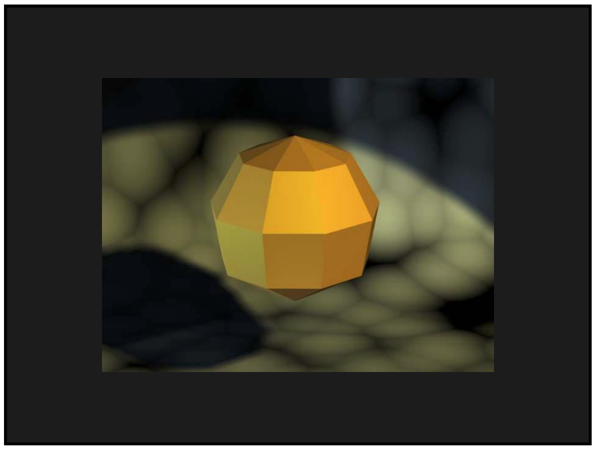
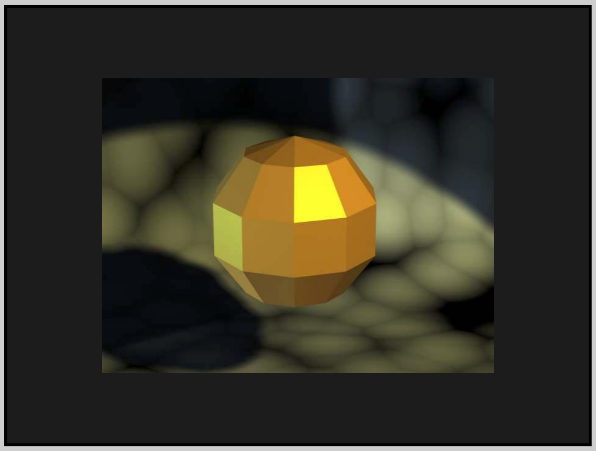
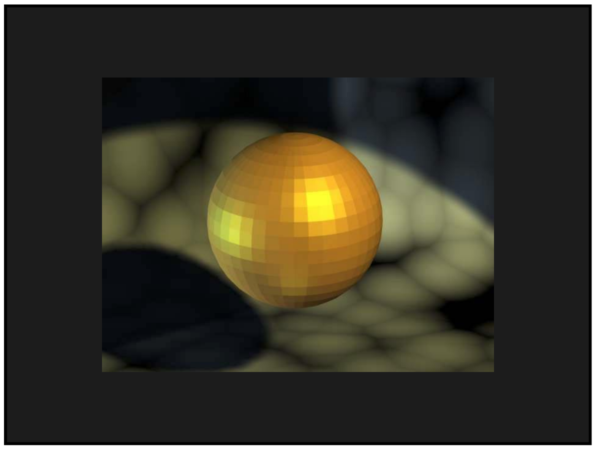
- Polygons
- Patches

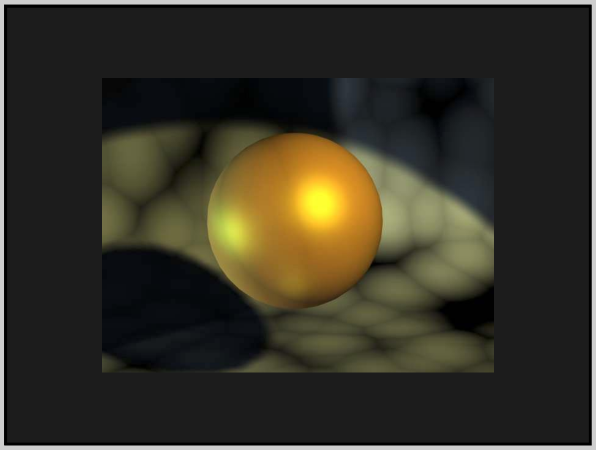


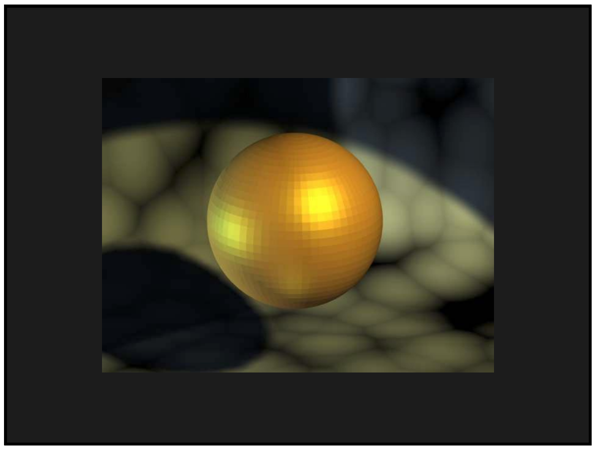
Polygons

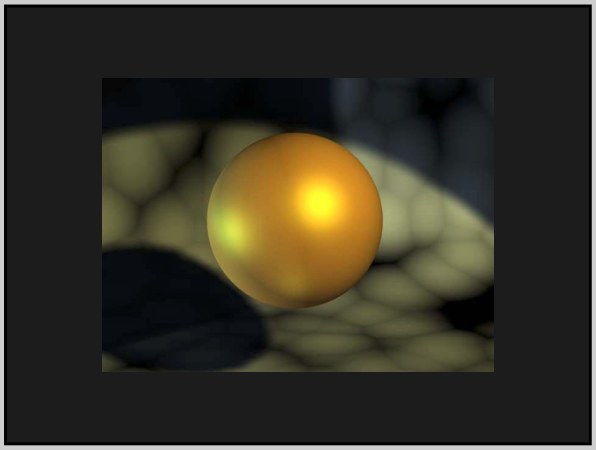
- Simple to define and use
- Assemble to make polyhedra
- Flat
- Flat
- Flat
- Really, really flat, always

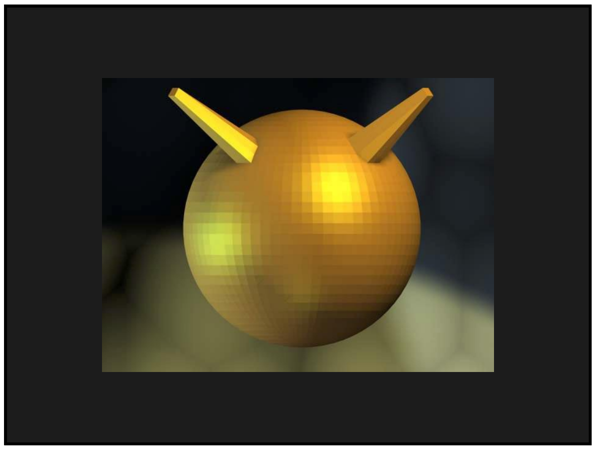


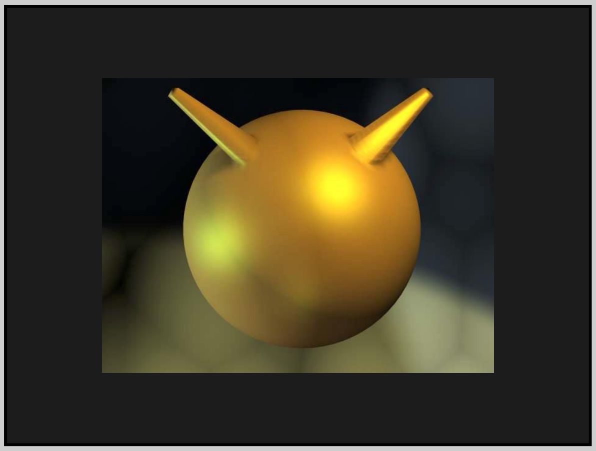







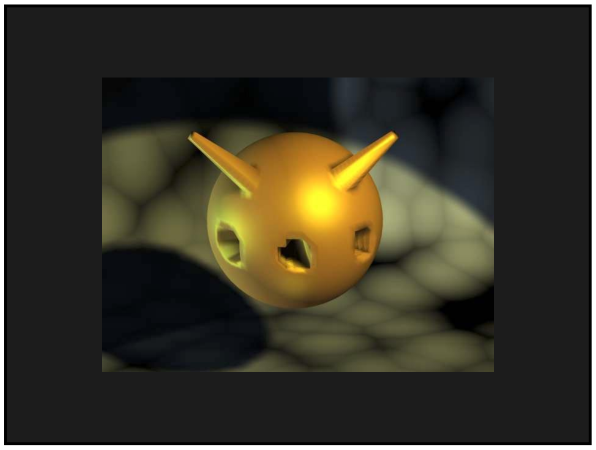


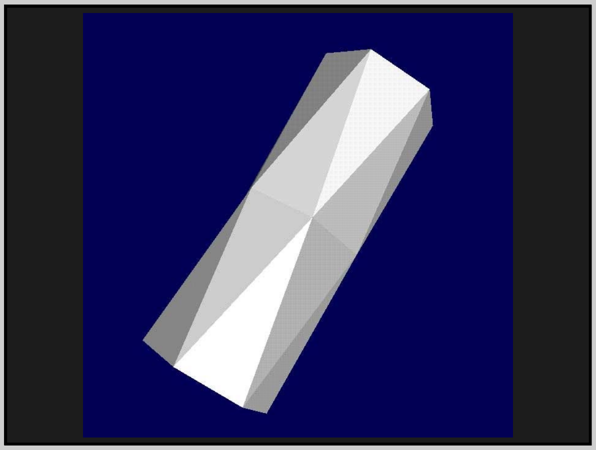


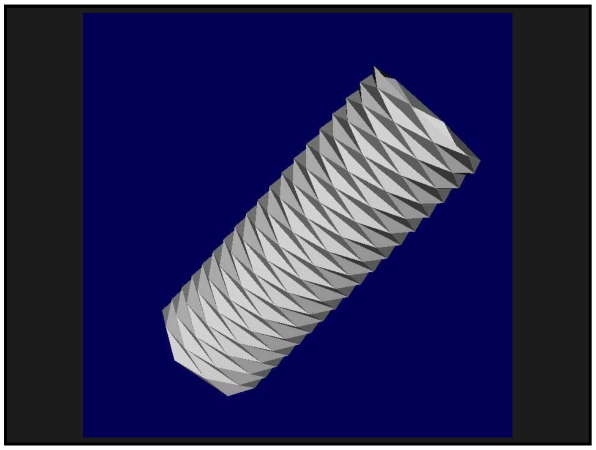


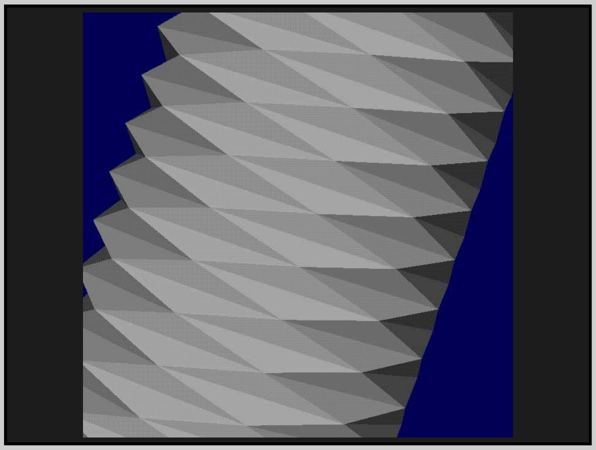


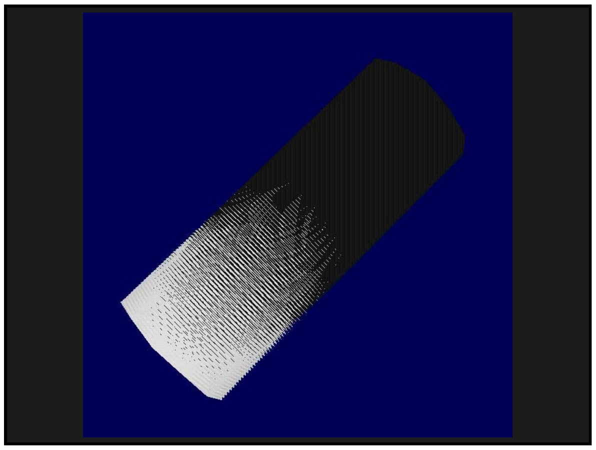


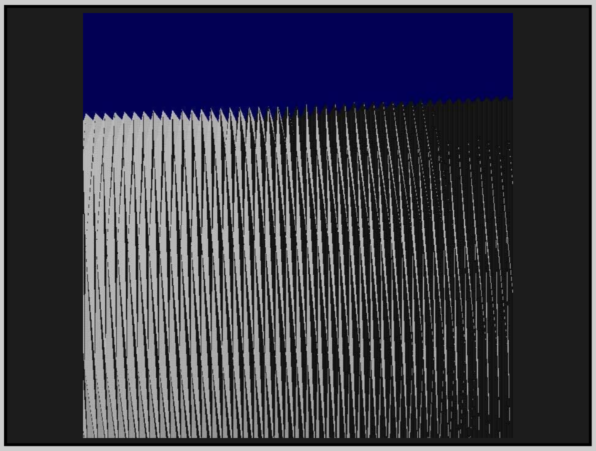


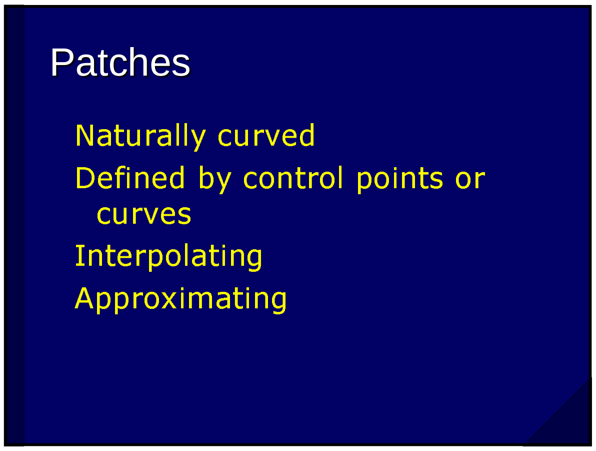






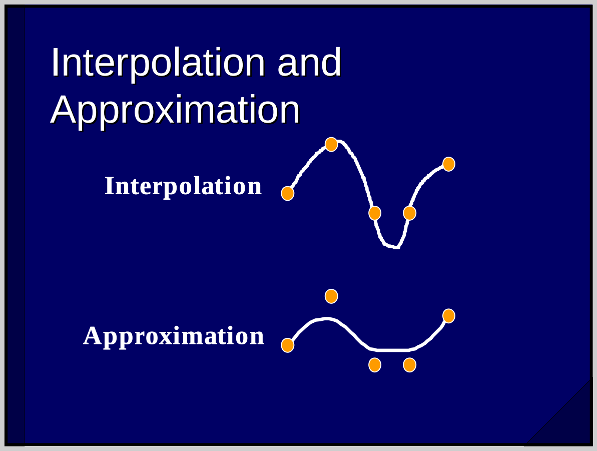




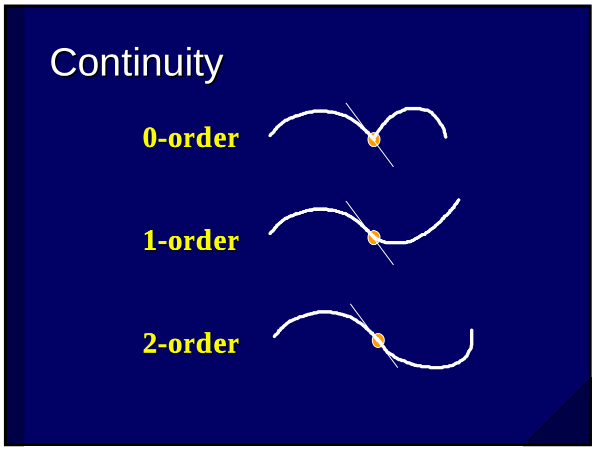


Patches

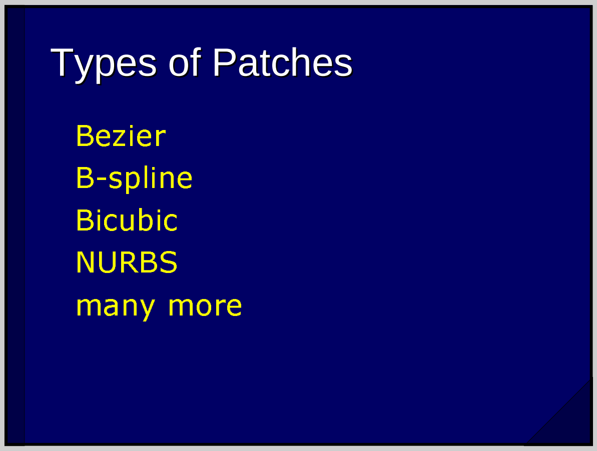
- Naturally curved
- Defined by control points or curves
- Interpolating
- Approximating



Interpolation and Approximation

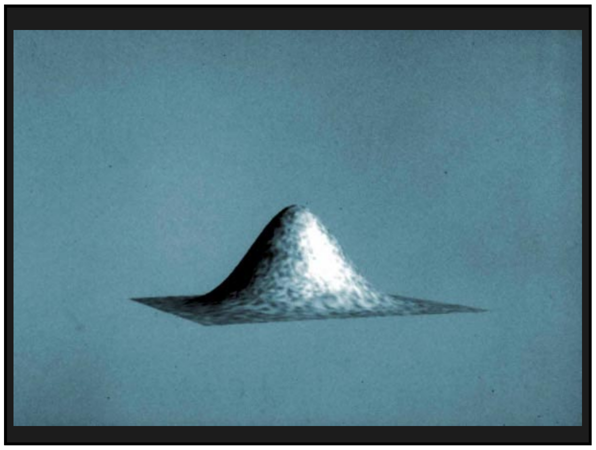


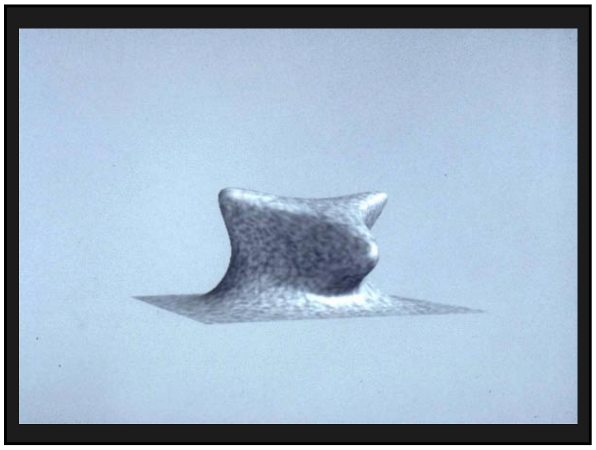
Continuity

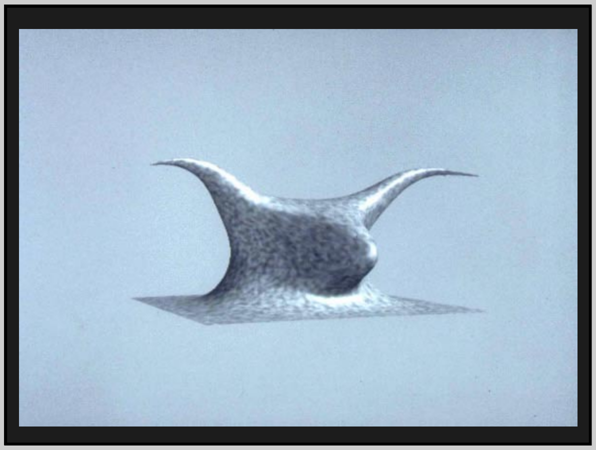


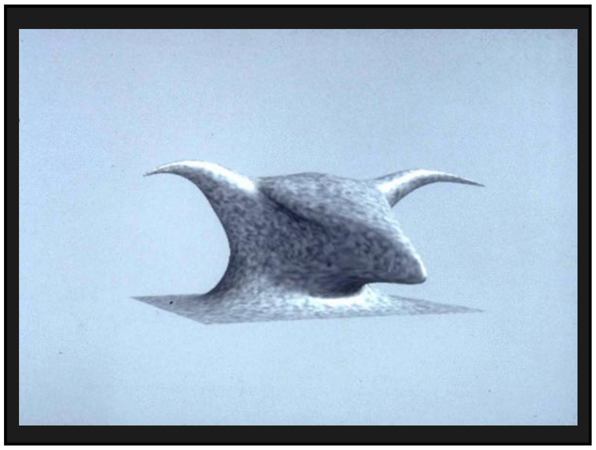
Types of Patches

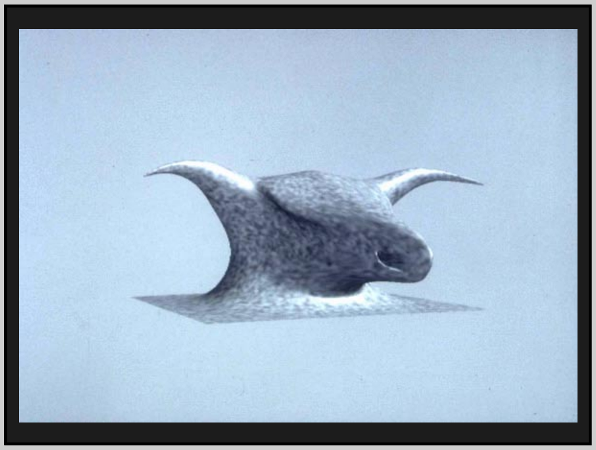
- Bezier
- B-spline
- Bicubic
- NURBS
- many more

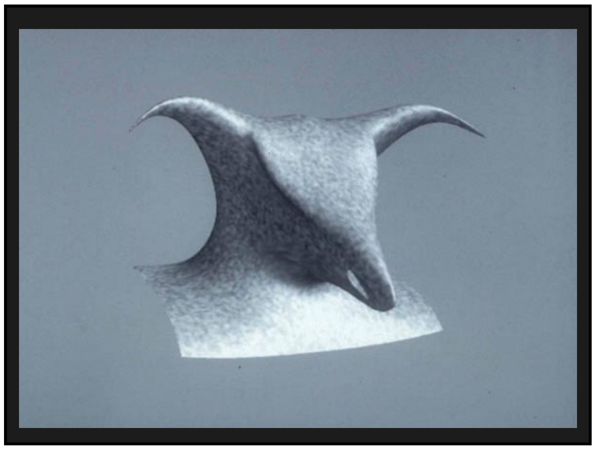


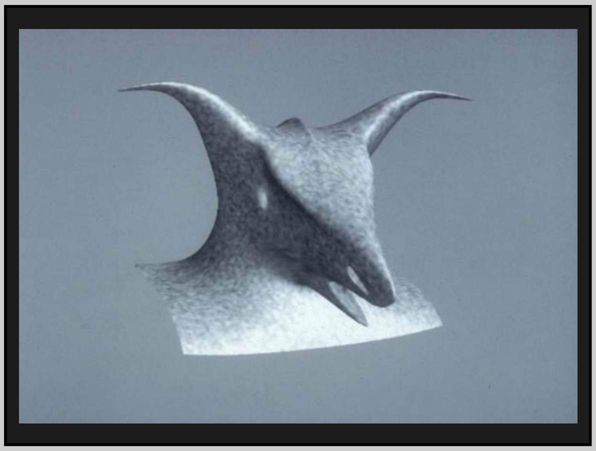


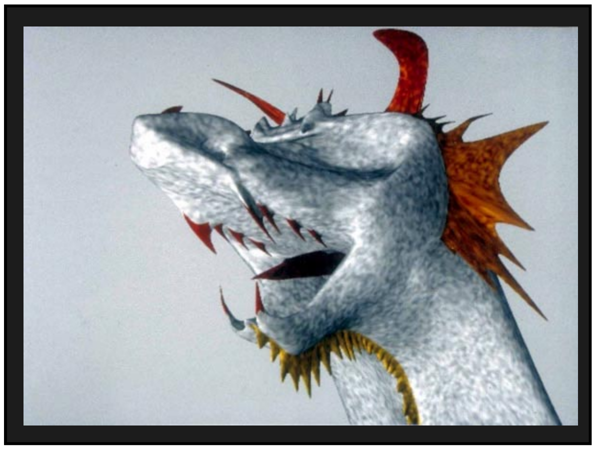


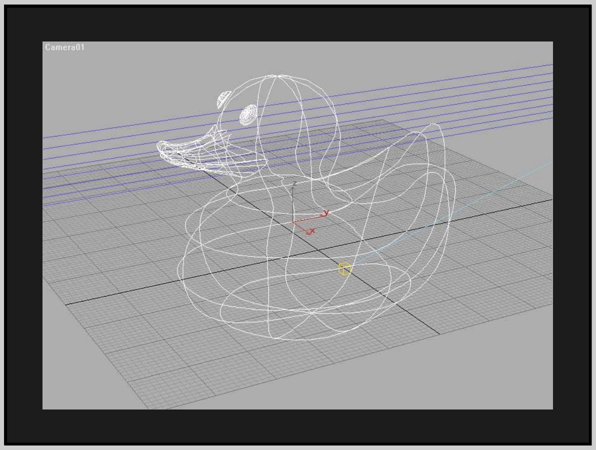


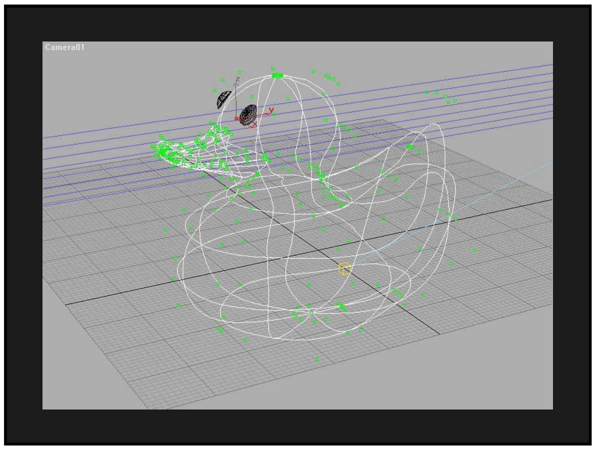


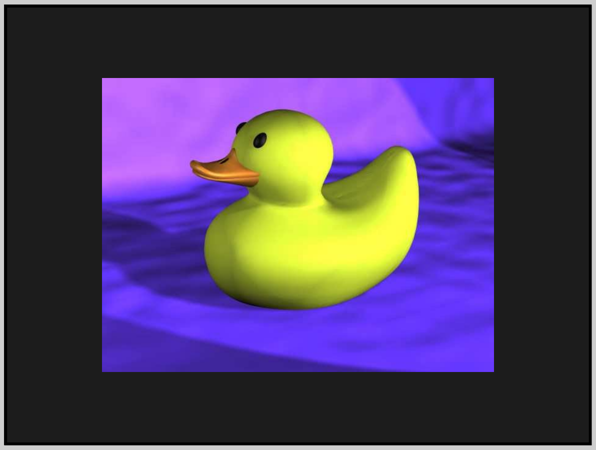


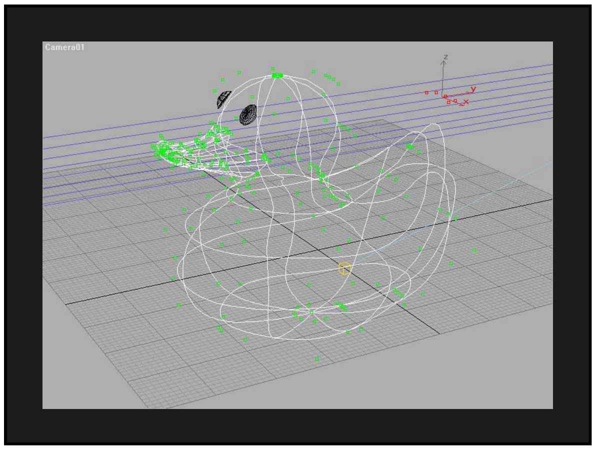


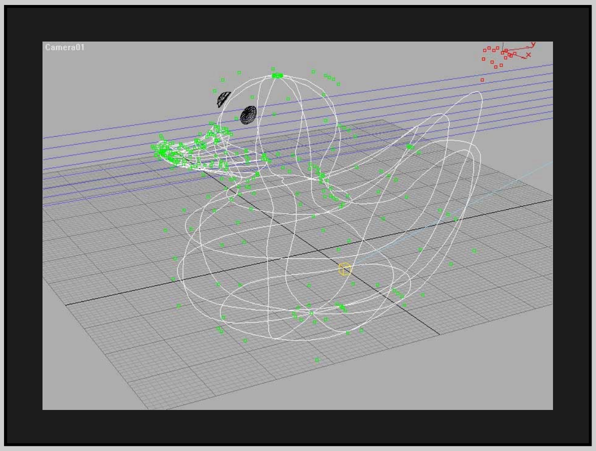


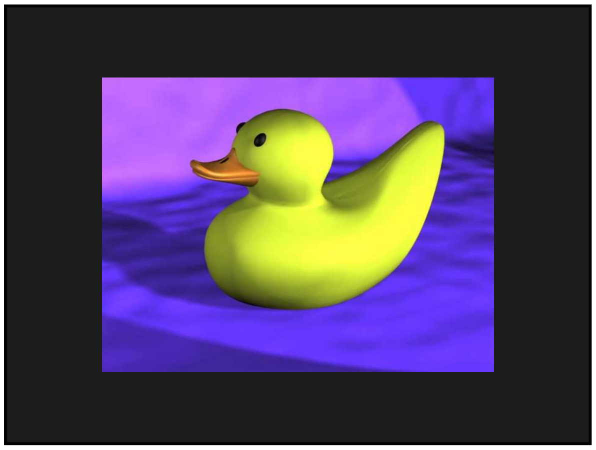

Camera01

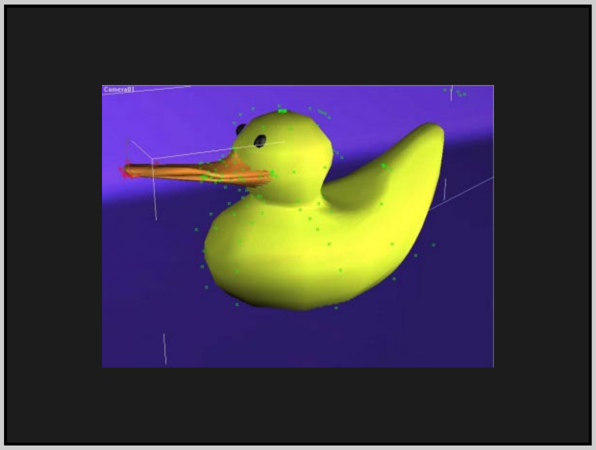

Camera01

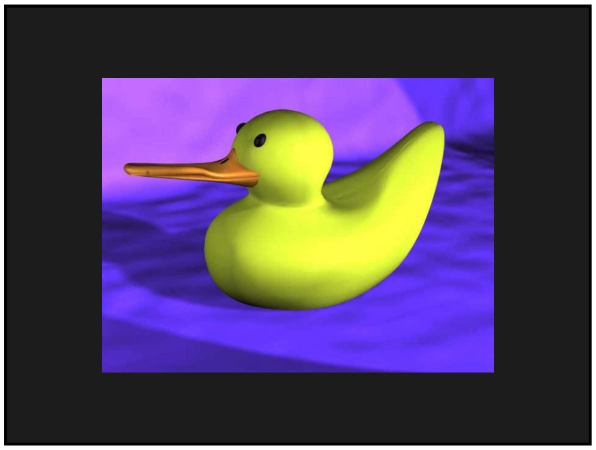


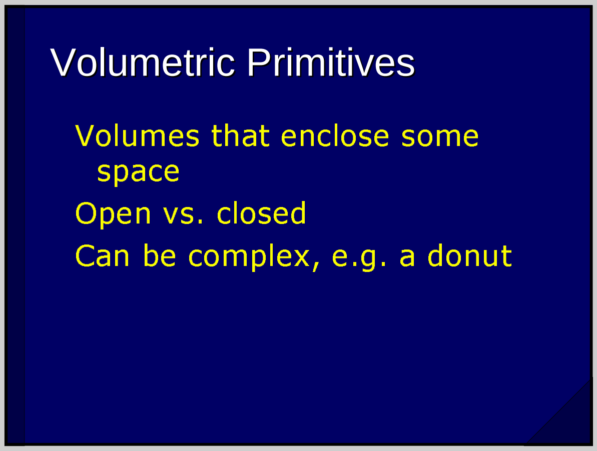

Camera01


Camera01



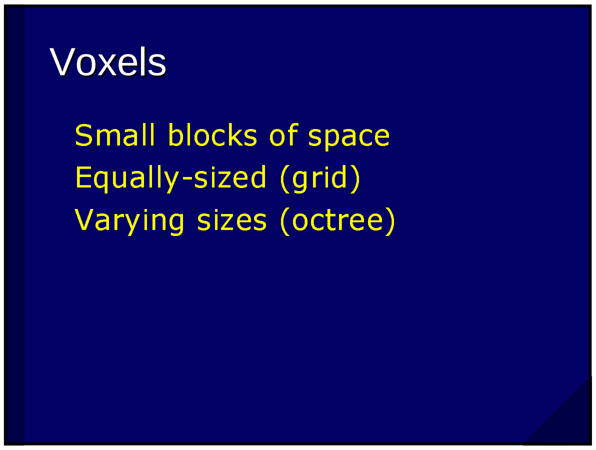






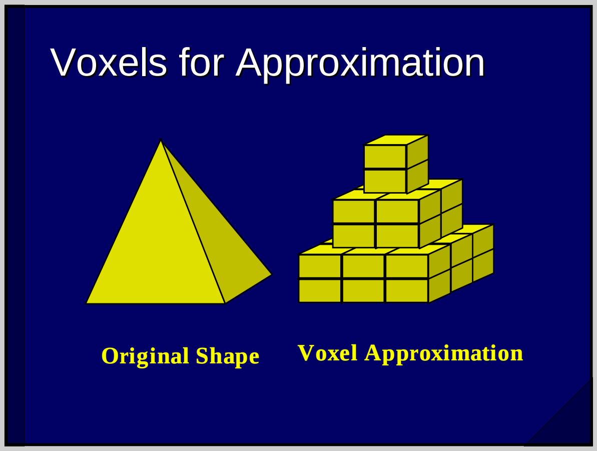
Volumetric Primitives

- Volumes that enclose some space
- Open vs. closed
- Can be complex, e.g. a donut



Voxels

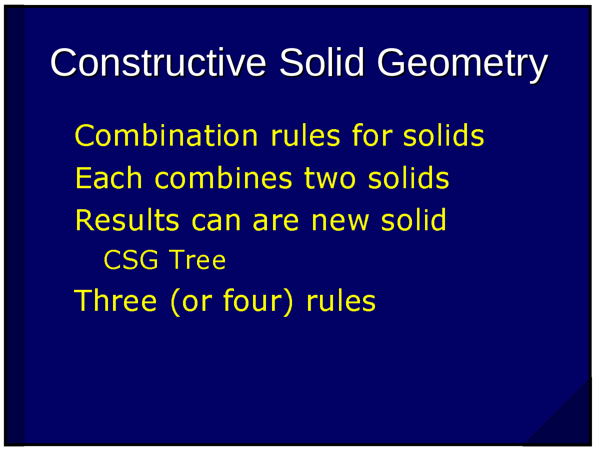
- Small blocks of space
- Equally-sized (grid)
- Varying sizes (octree)



Voxels for Approximation

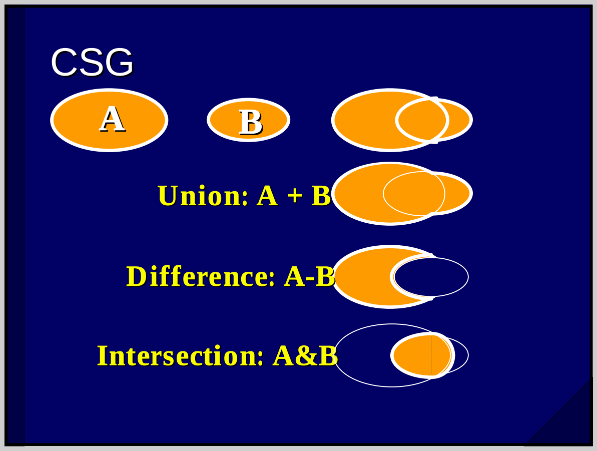
Original Shape

Voxel Approximation

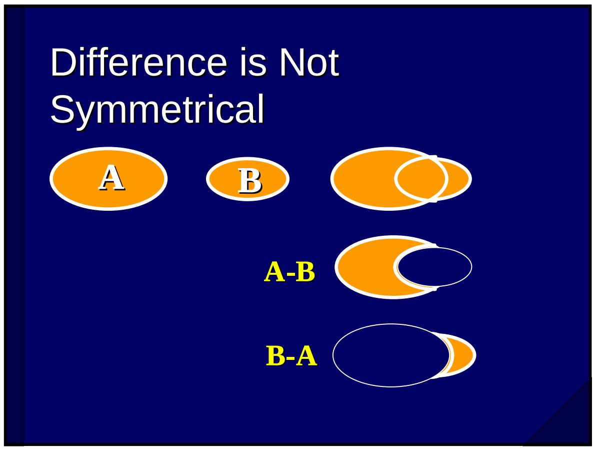


Constructive Solid Geometry

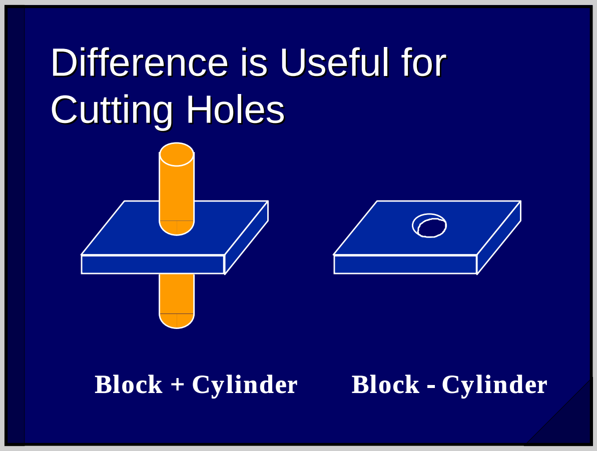
- Combination rules for solids
- Each combines two solids
- Results can are new solid
  - CSG Tree
- Three (or four) rules



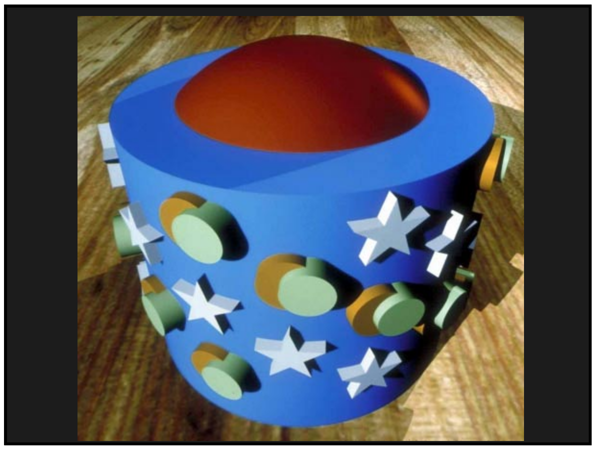
CSG

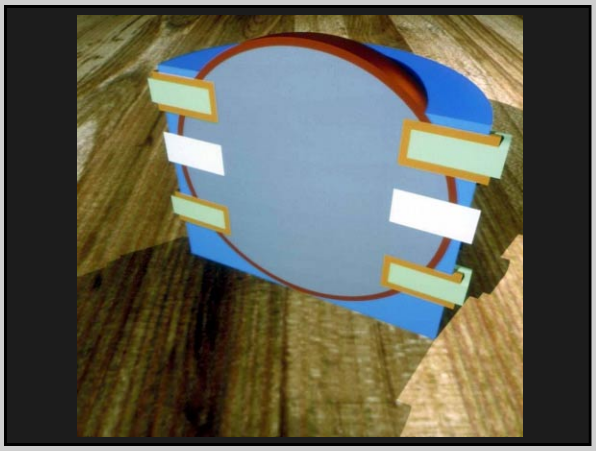


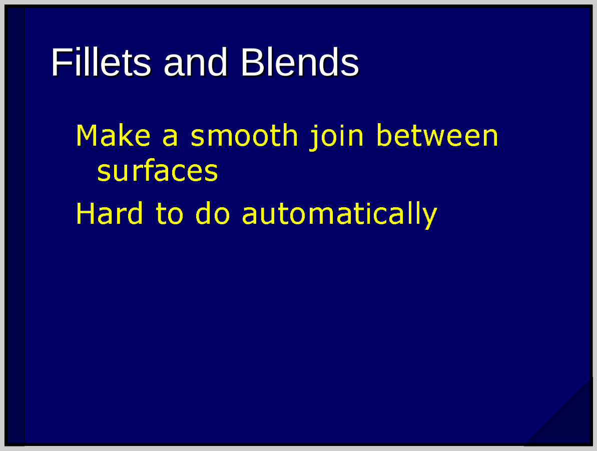
Difference is Not Symmetrical



Difference is Useful for Cutting Holes

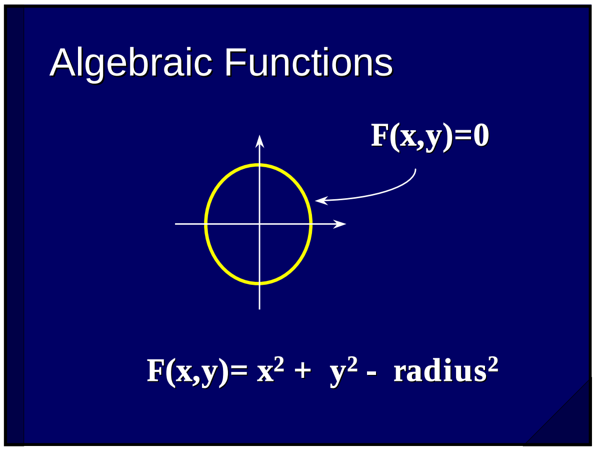




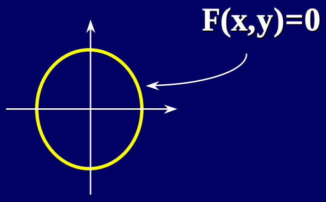


Fillets and Blends

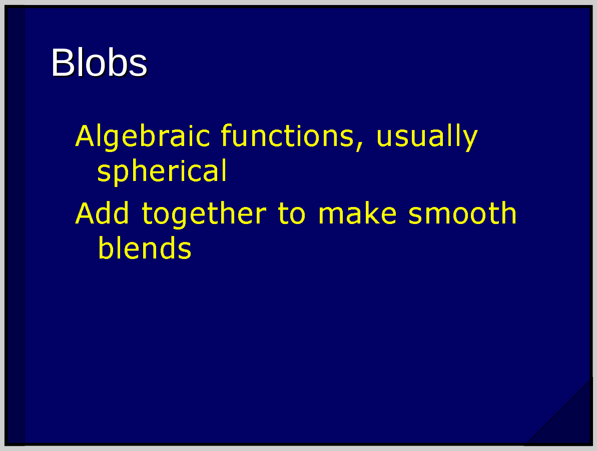
- Make a smooth join between surfaces
- Hard to do automatically



Algebraic Functions

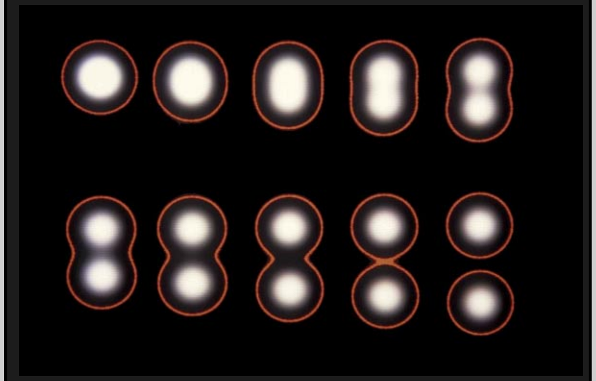


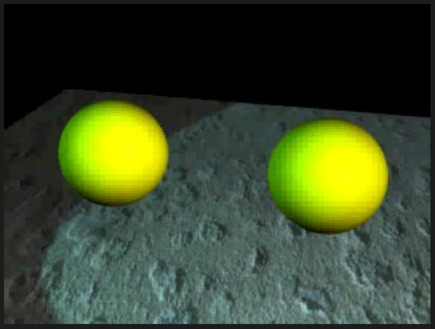
$F(x,y)= x^2 + y^2 - radius^2$

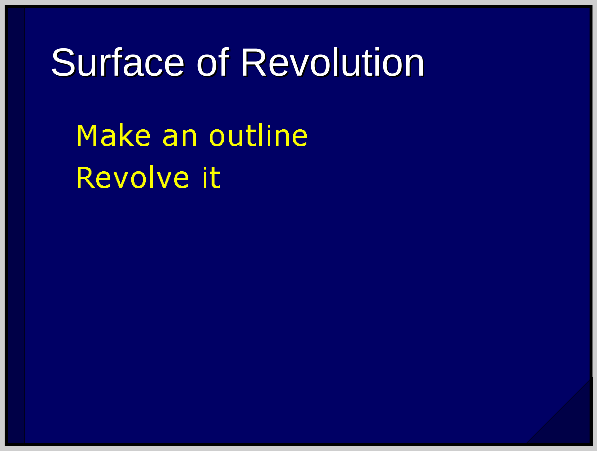


Blobs

- Algebraic functions, usually spherical
- Add together to make smooth blends

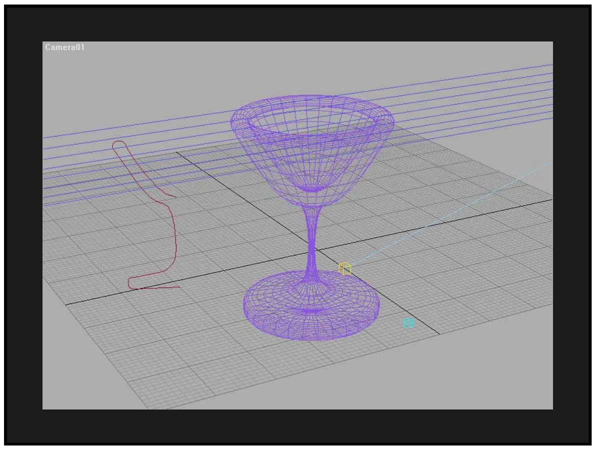


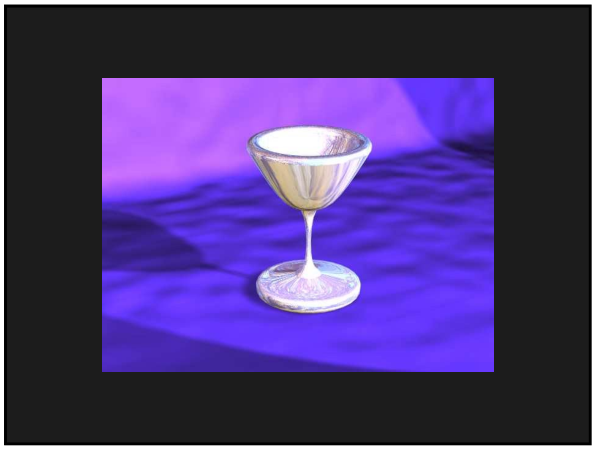


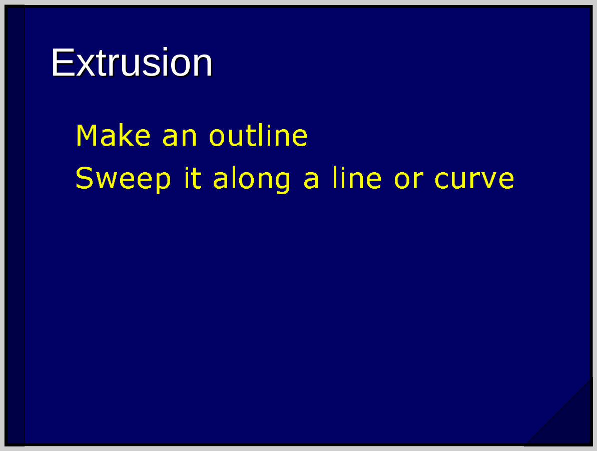


Surface of Revolution

- Make an outline
- Revolve it

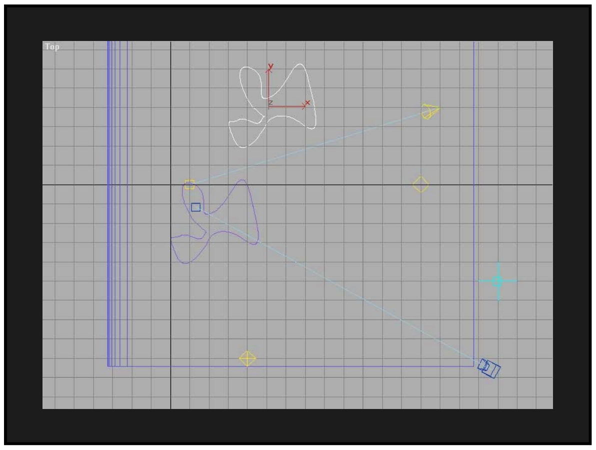


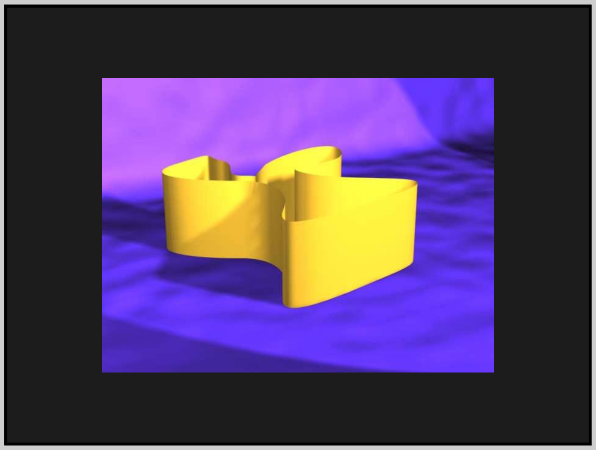


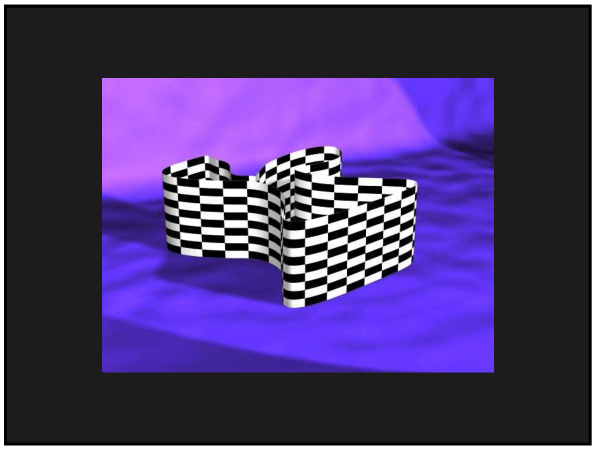


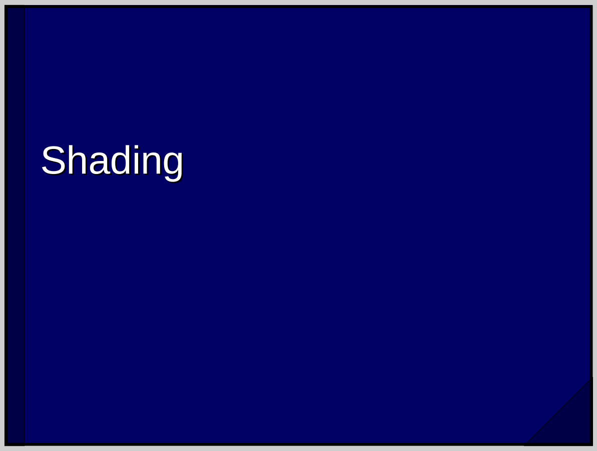
Extrusion

- Make an outline
- Sweep it along a line or curve

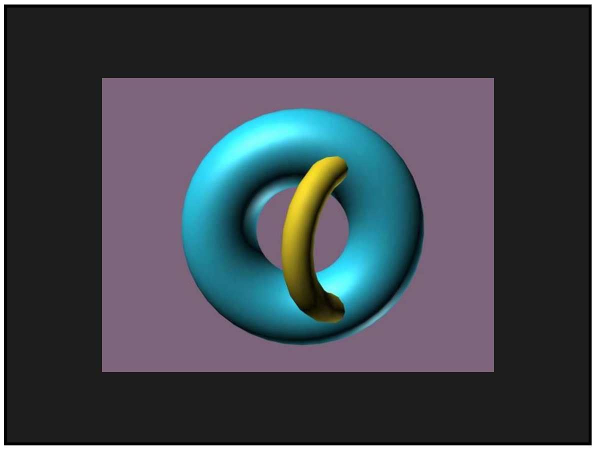


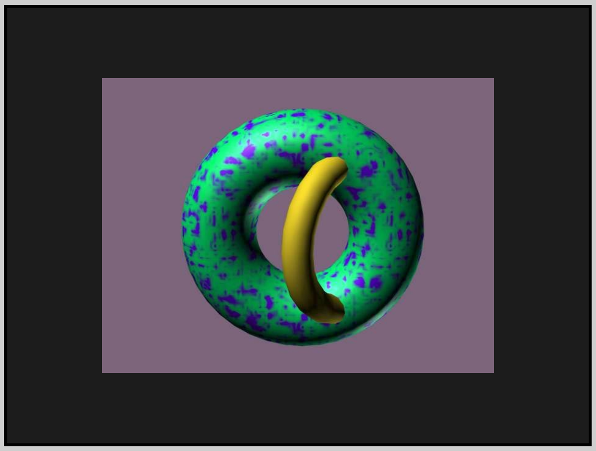


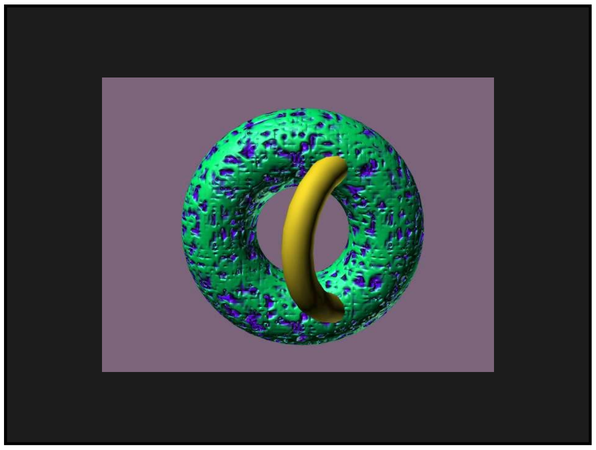


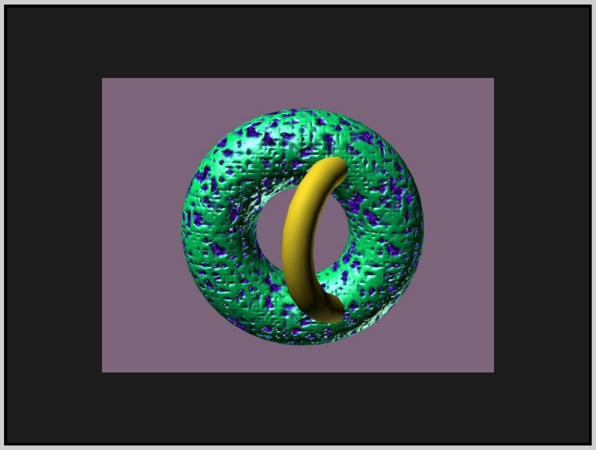


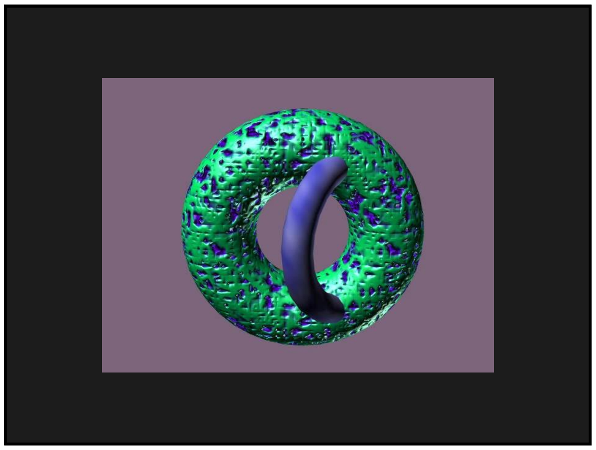
Shading

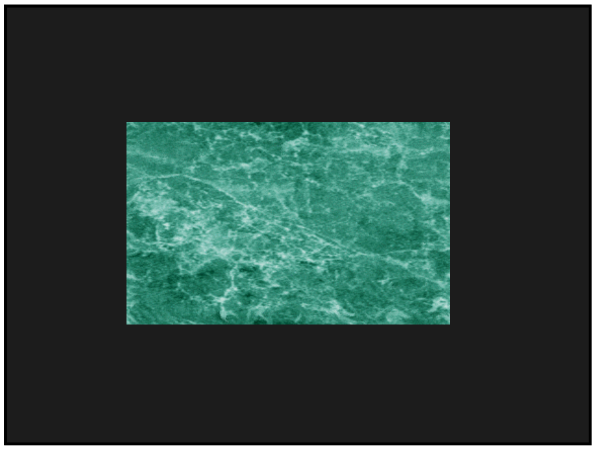


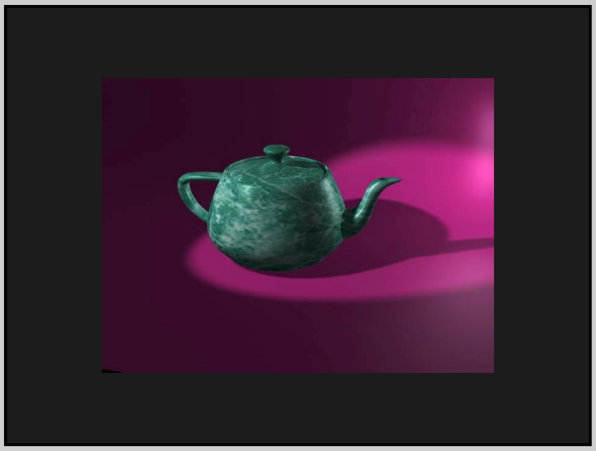


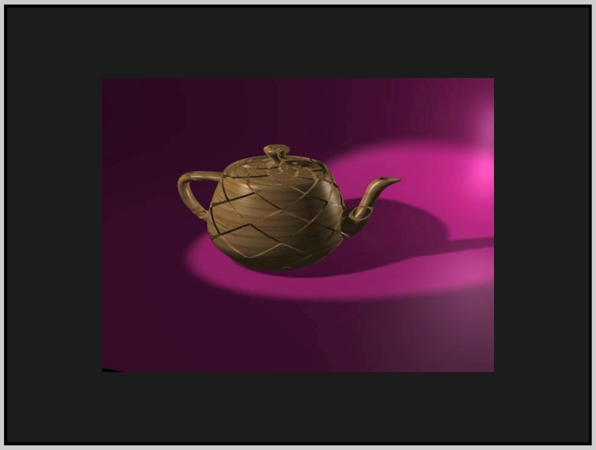
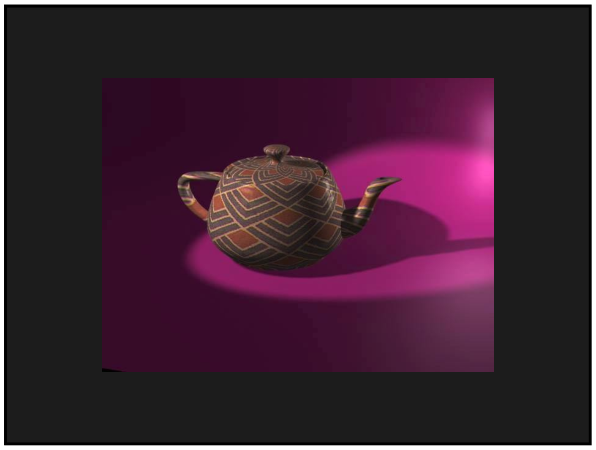
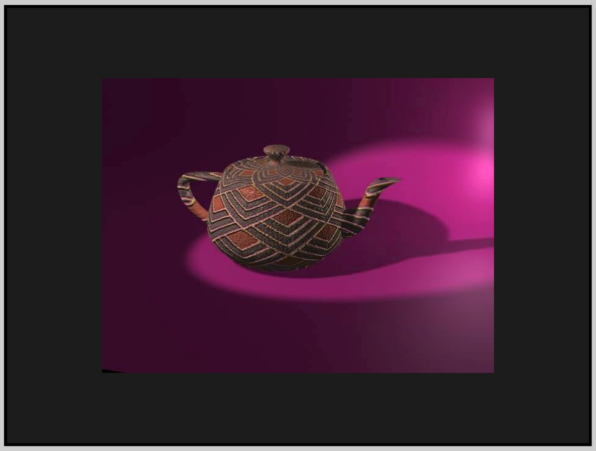
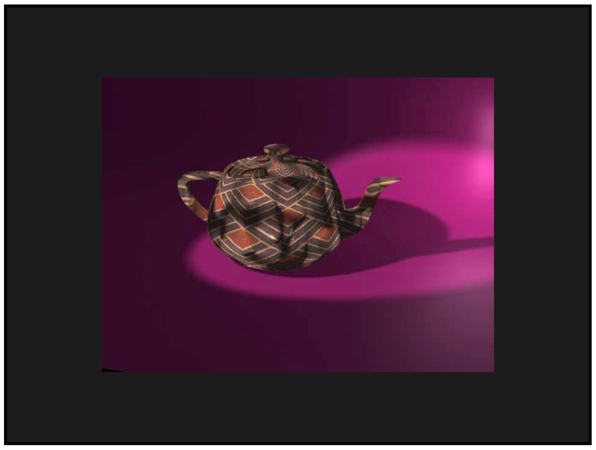


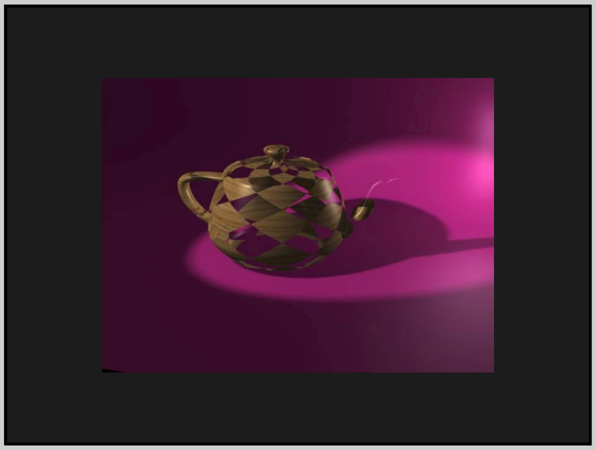


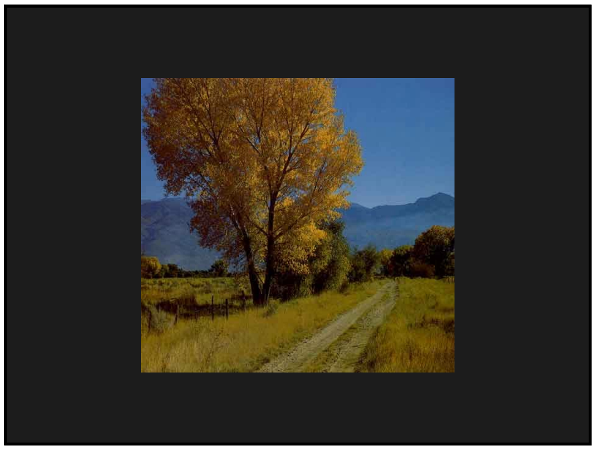


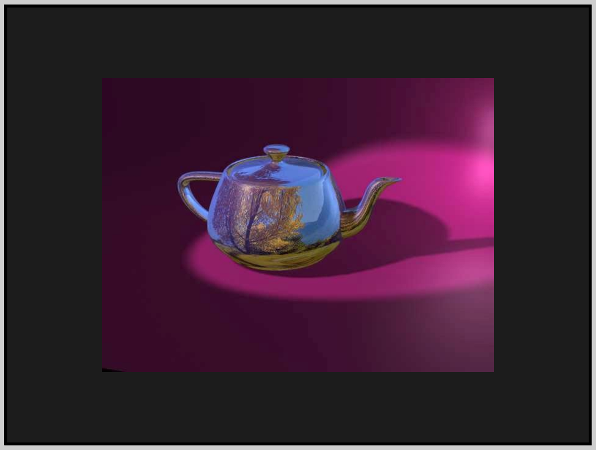




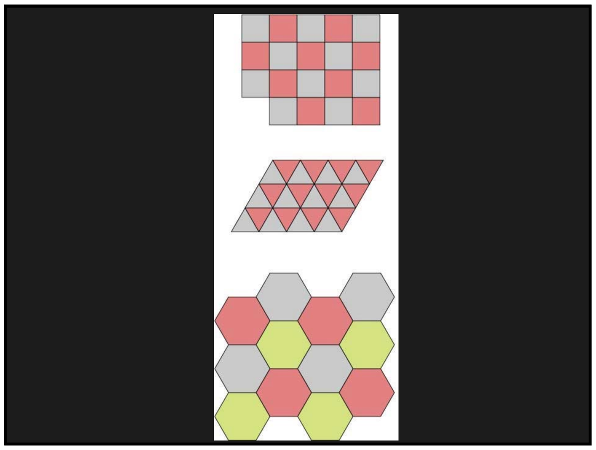





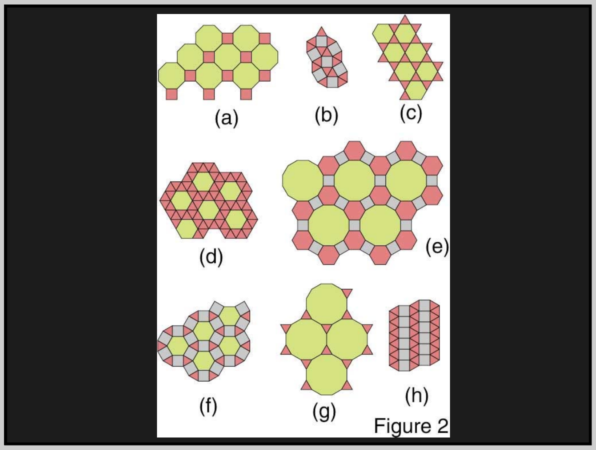


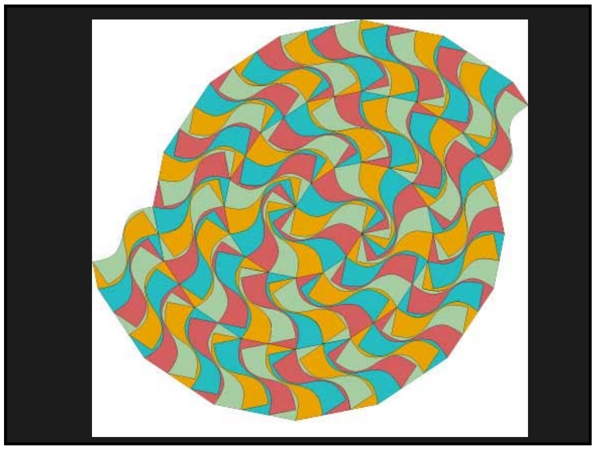


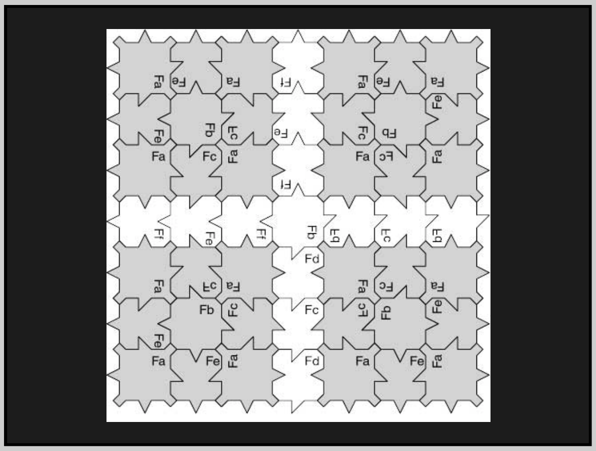


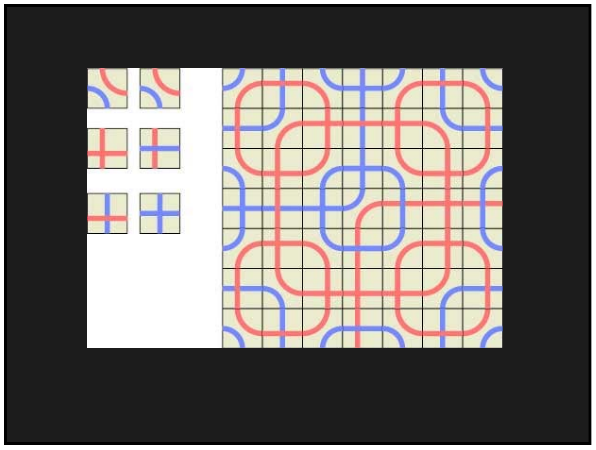
Texture

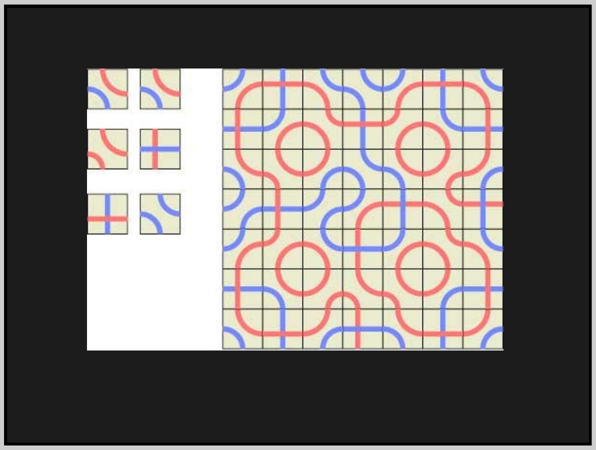


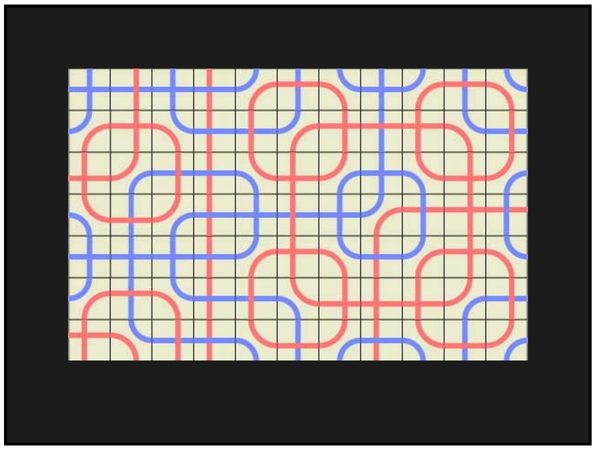

(a)
(b)
(c)
(d)
(e)
(f)
(g)
(h)
Figure 2

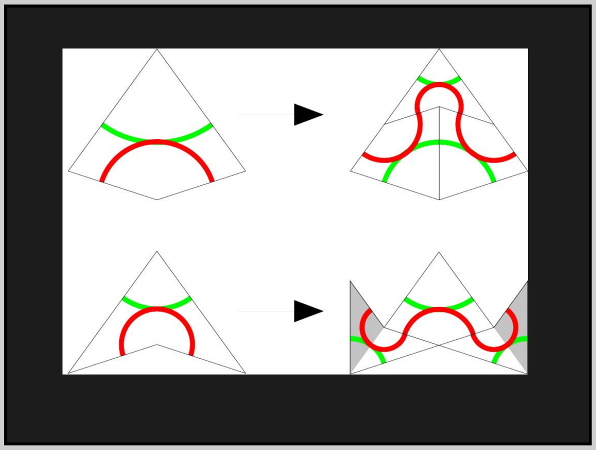


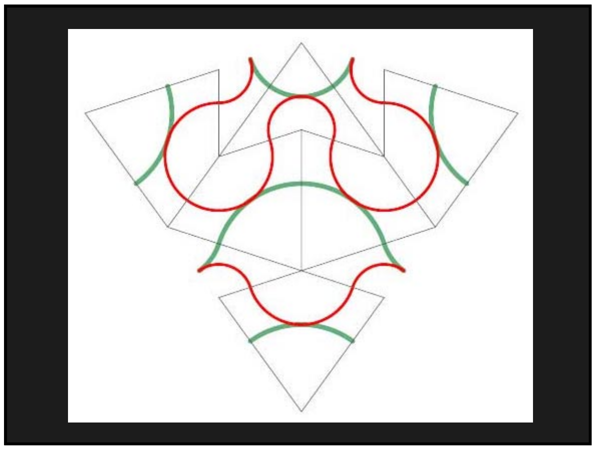


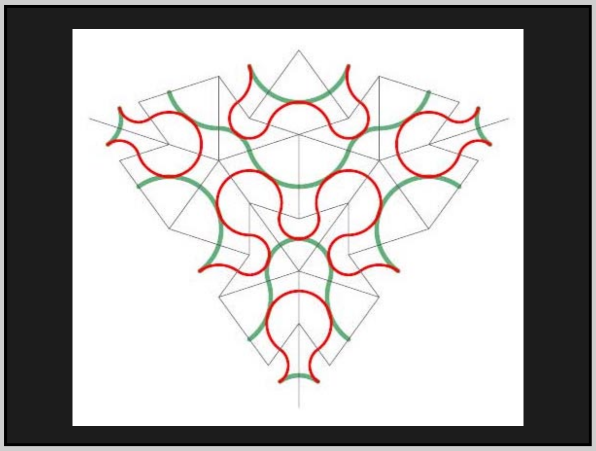


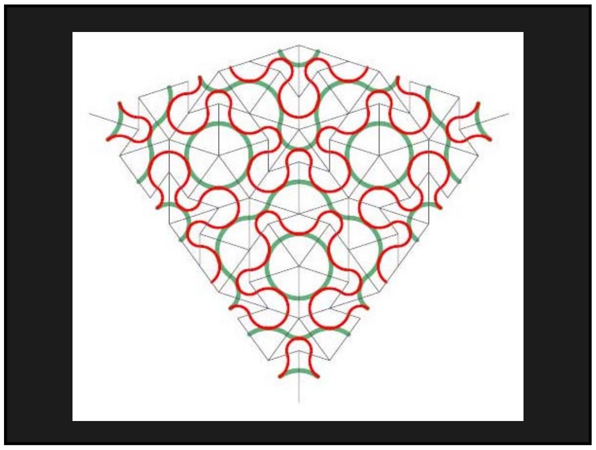


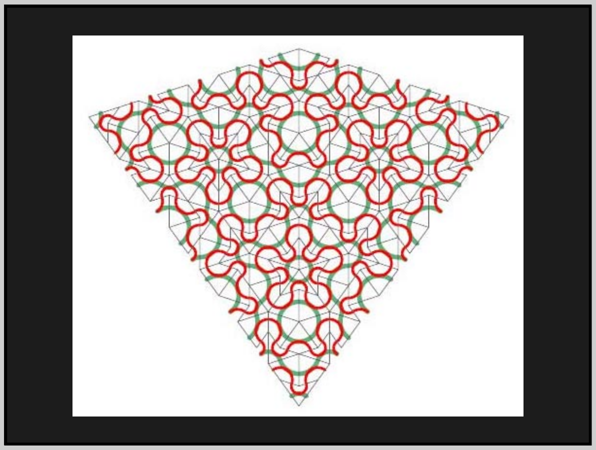


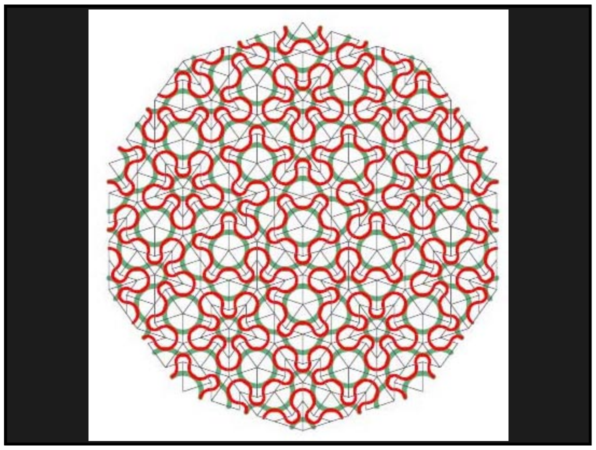


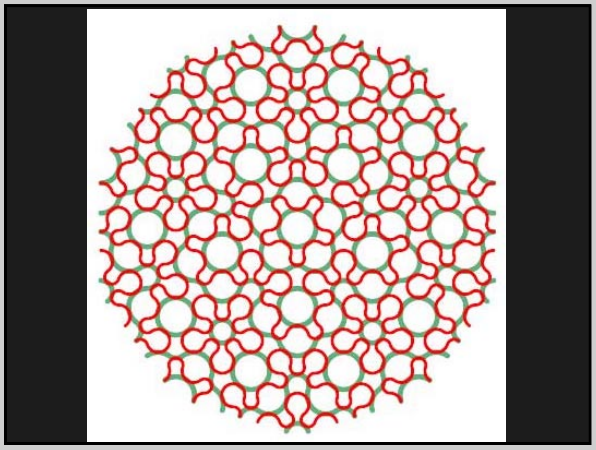


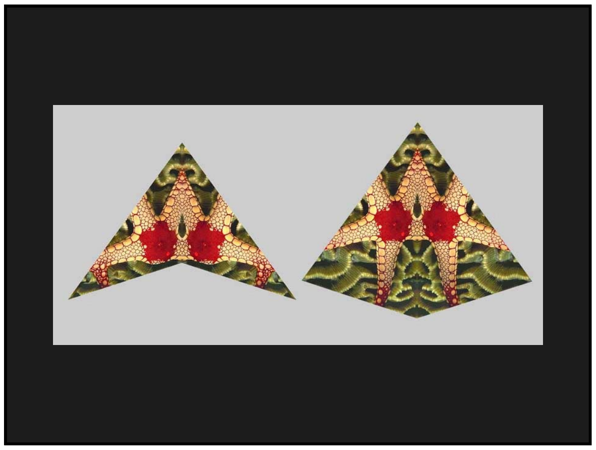


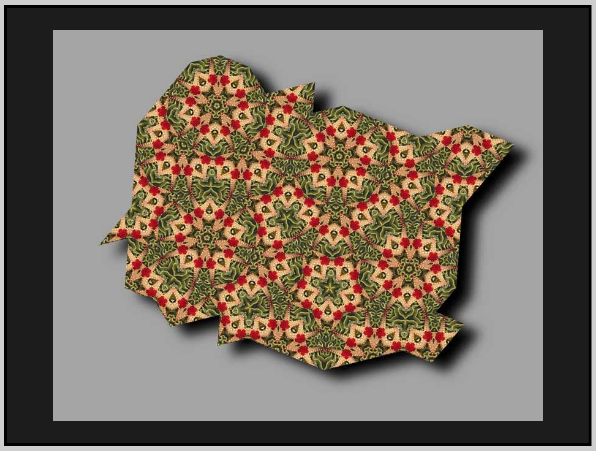


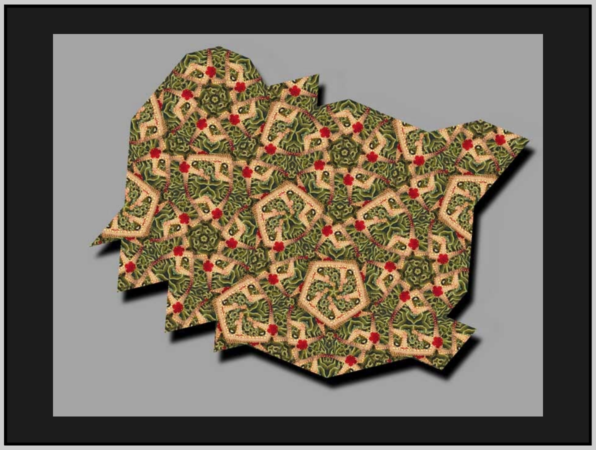


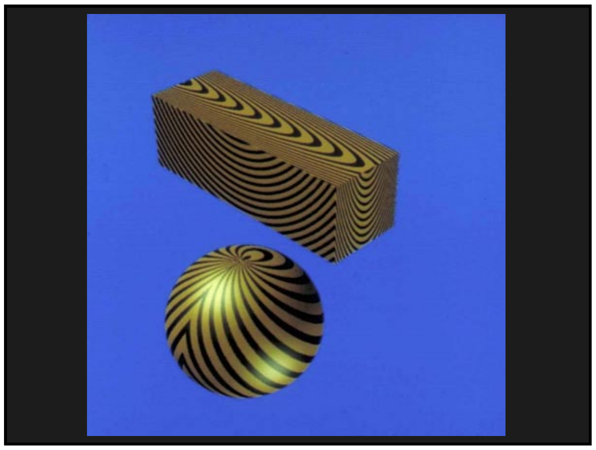


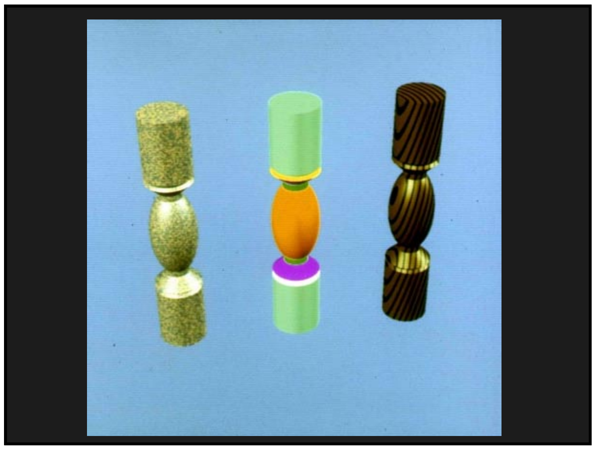


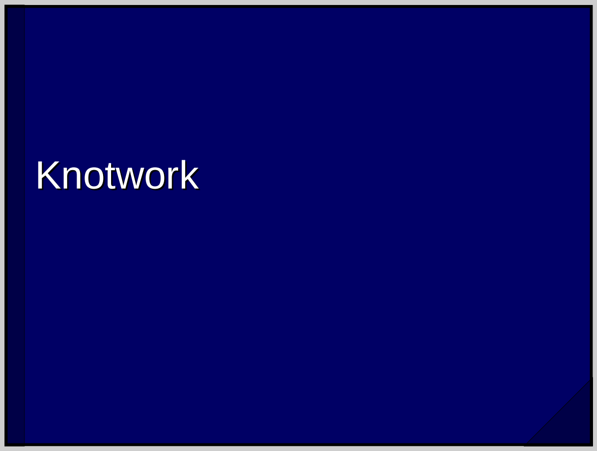




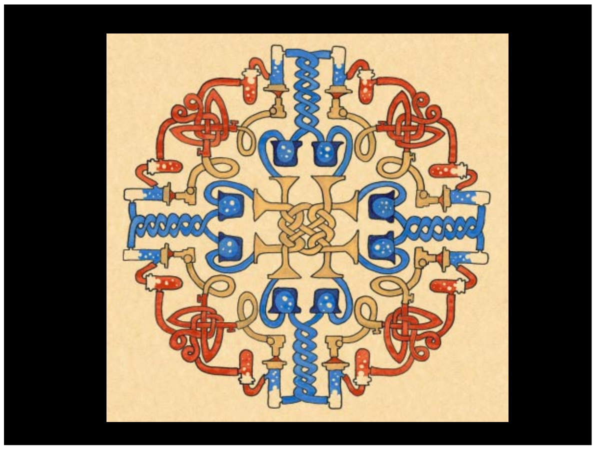


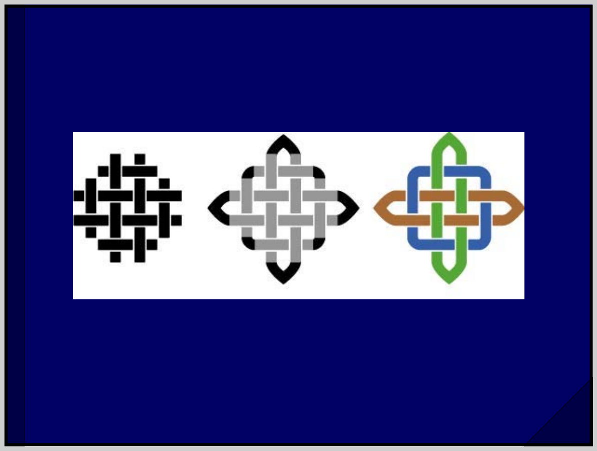


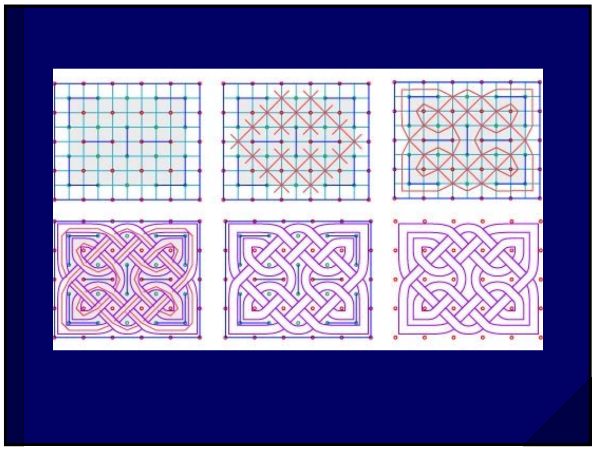


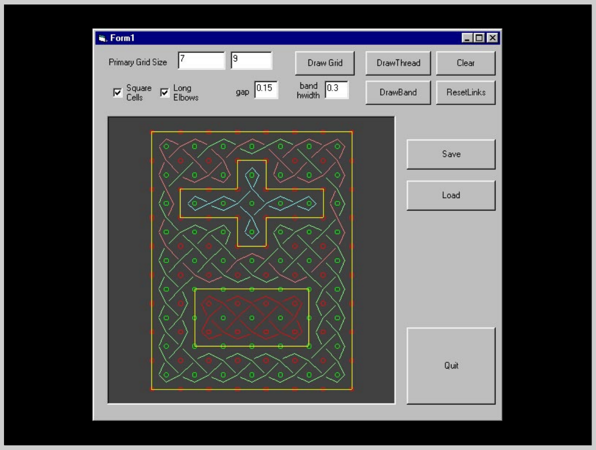


Knotwork



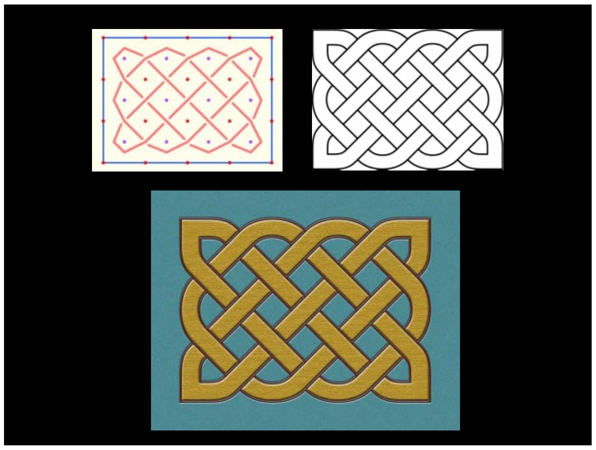


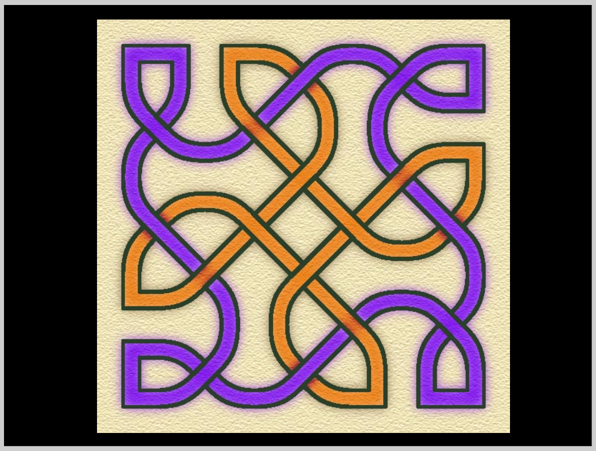


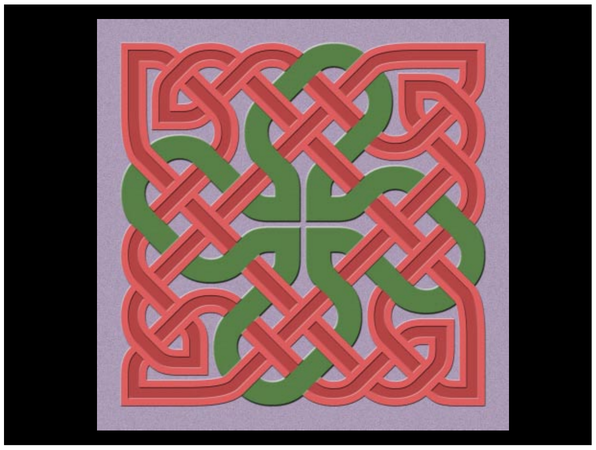


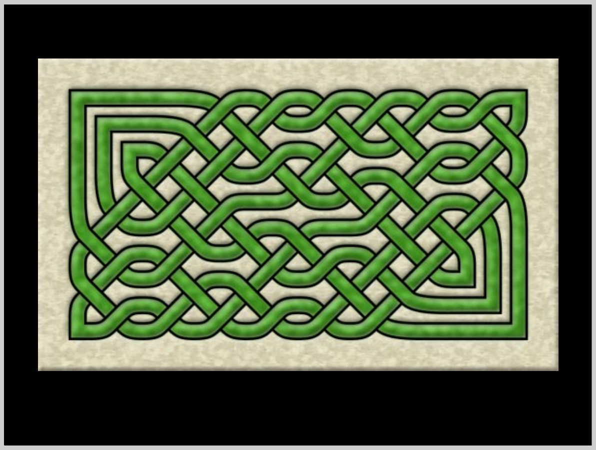
Primary Grid Size 7 9

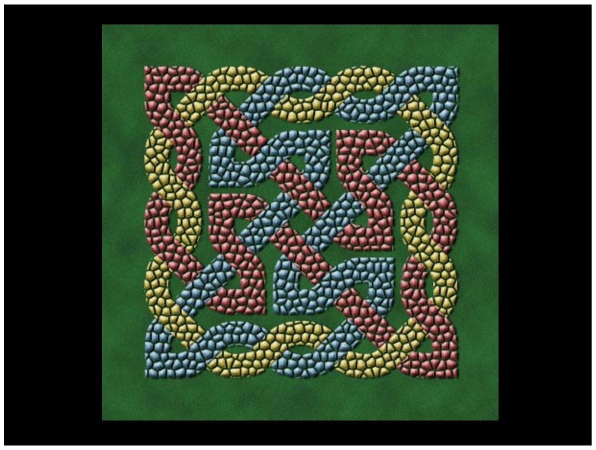
[✓ Square Cells] [✓ Long Elbows] gap 0.15 band hwidth 0.3

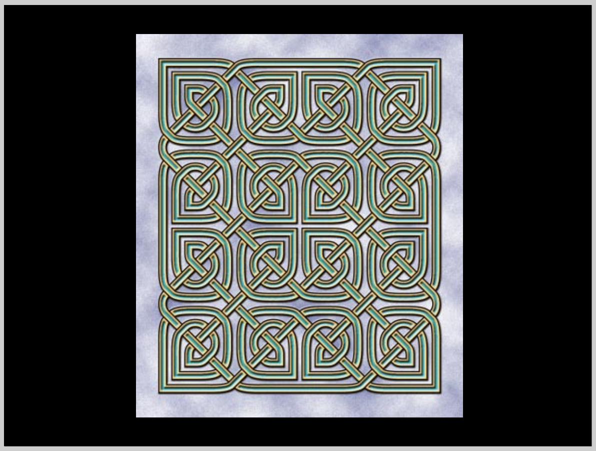


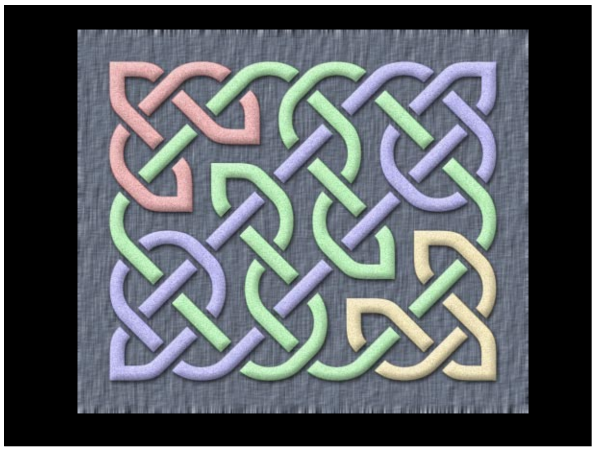


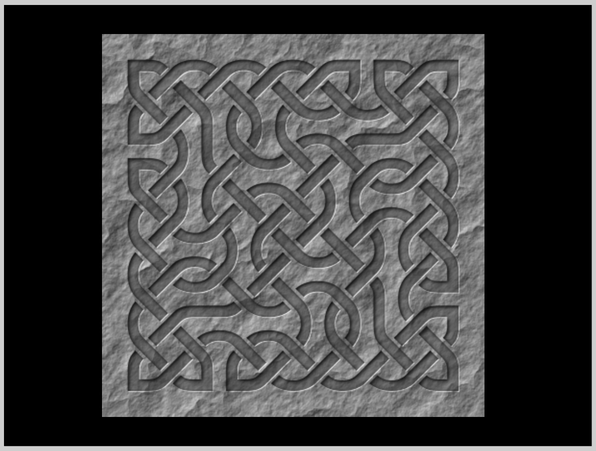


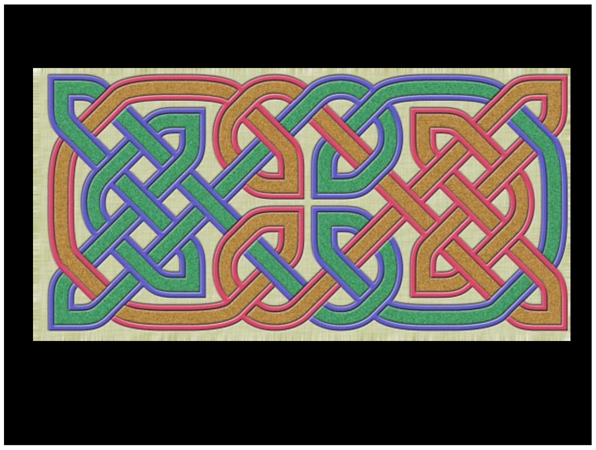


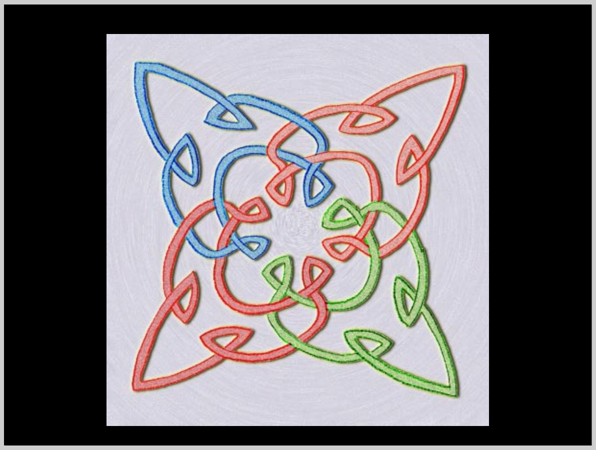


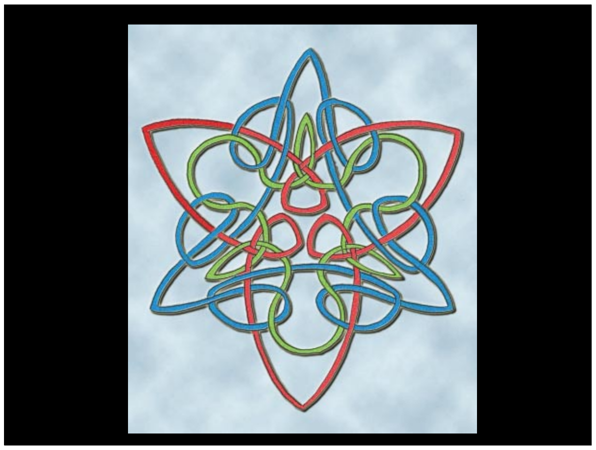


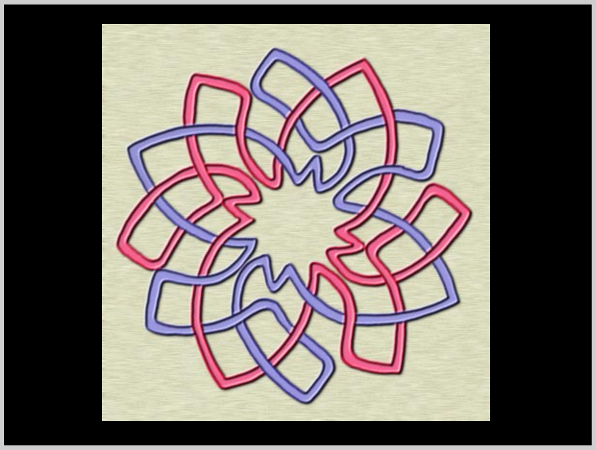


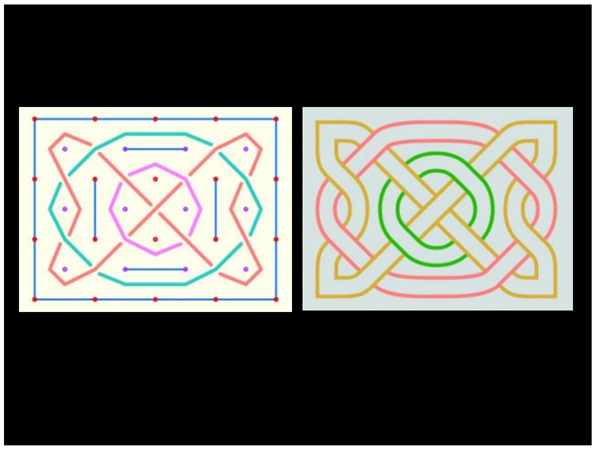


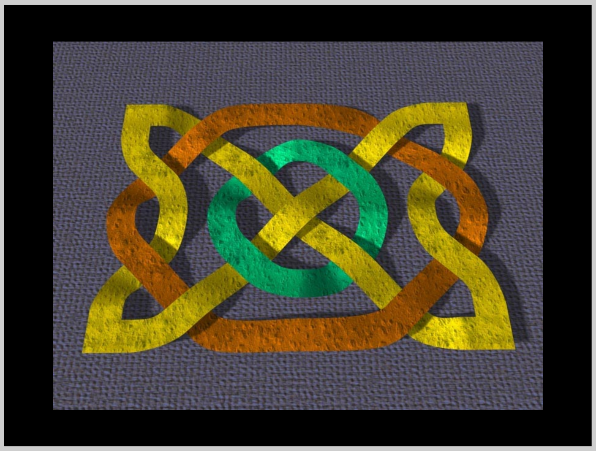


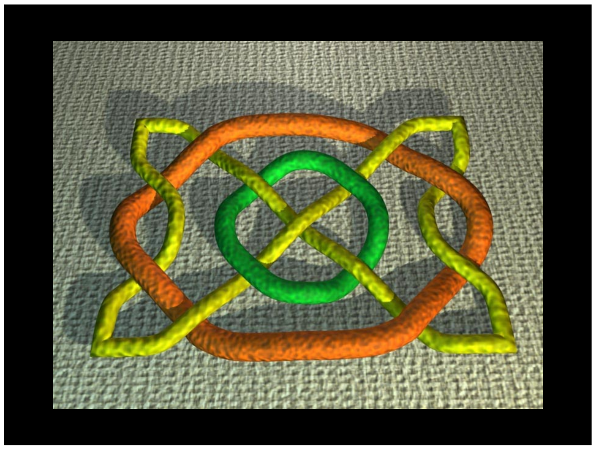


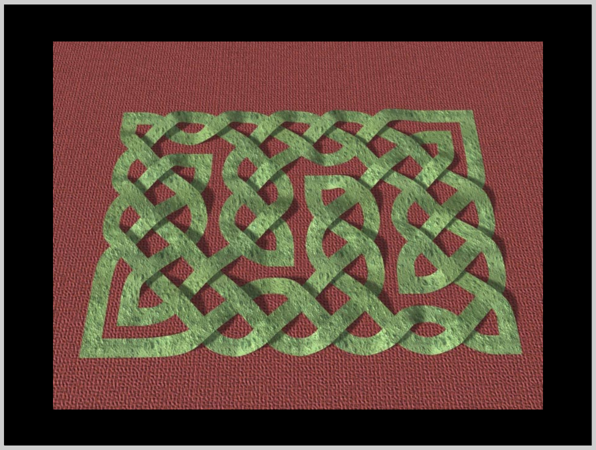
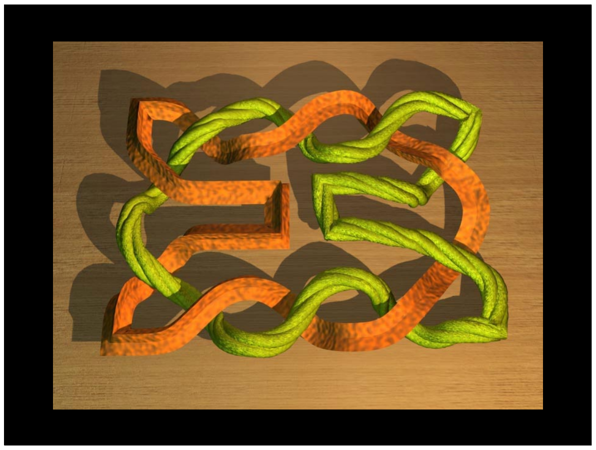
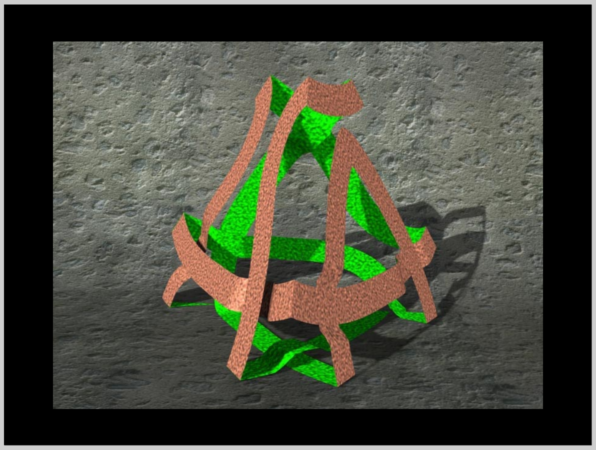
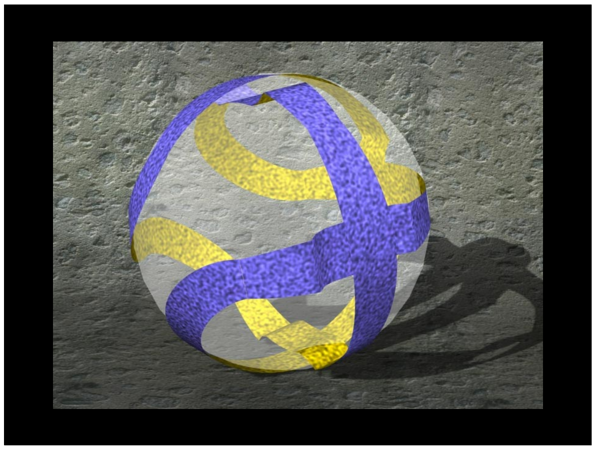
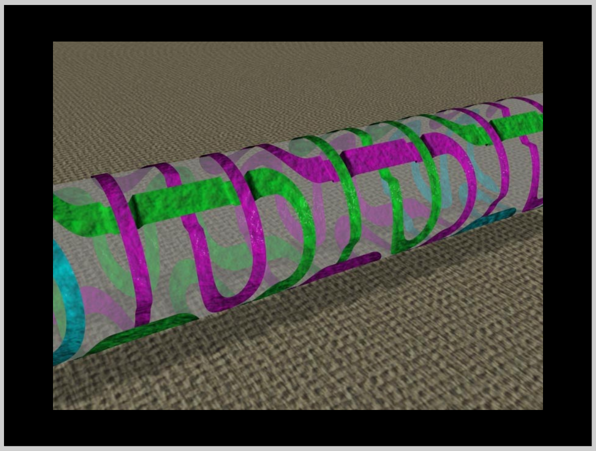
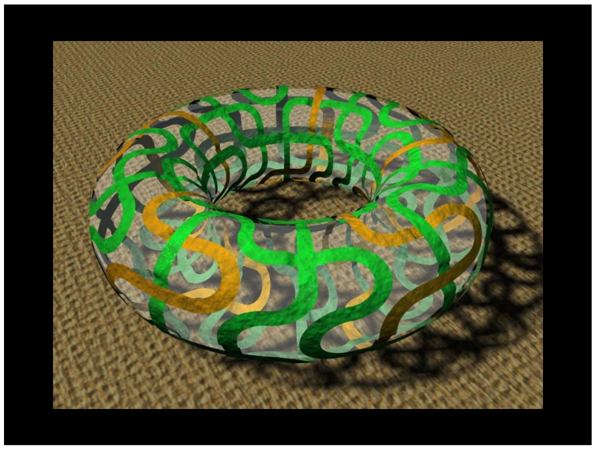


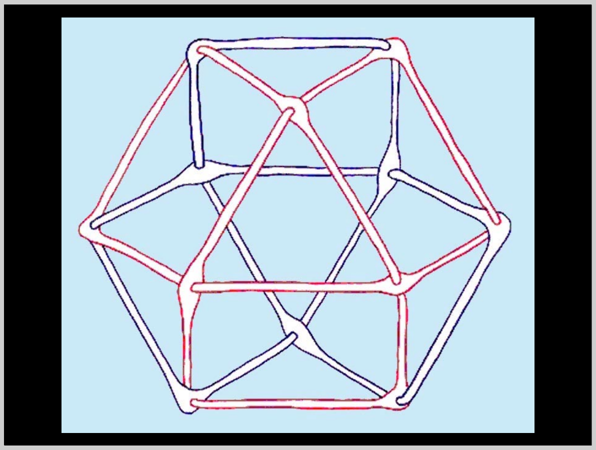


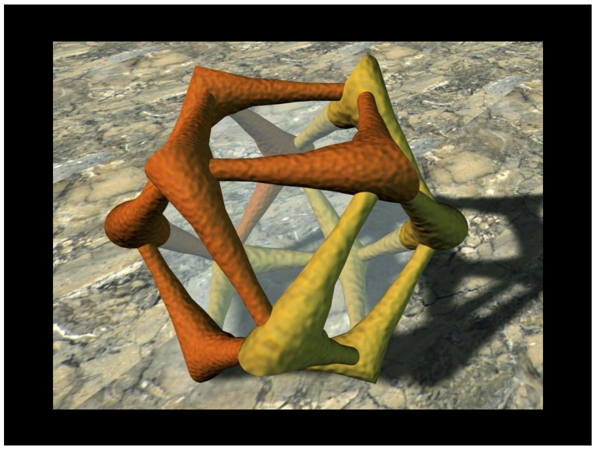


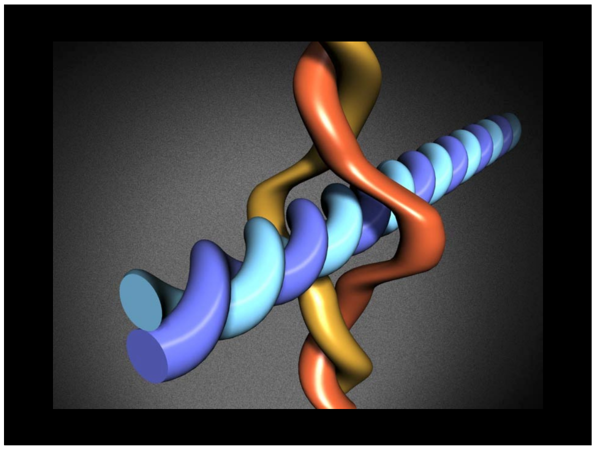


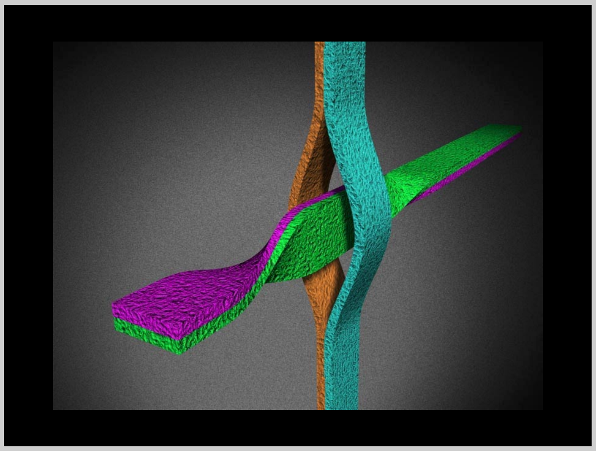


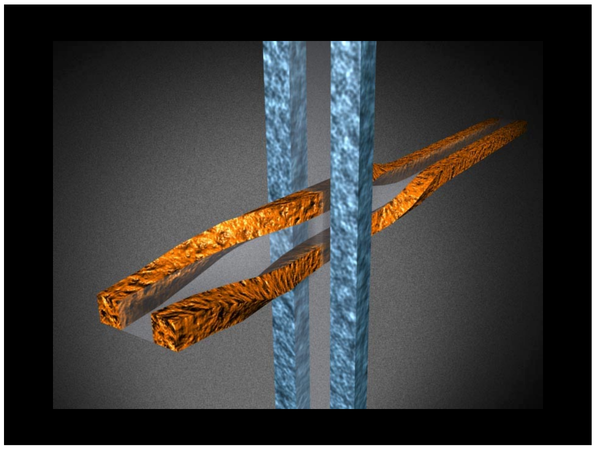







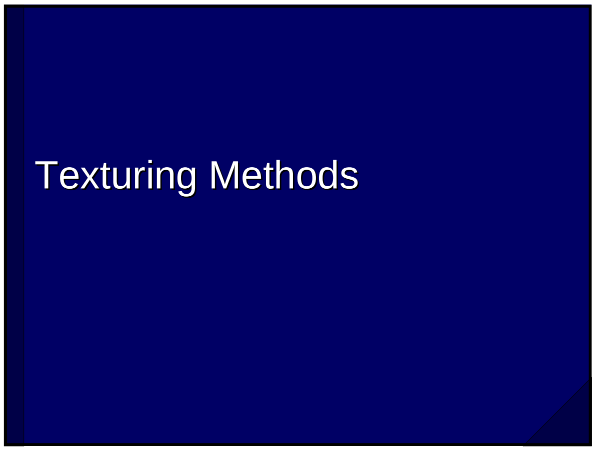




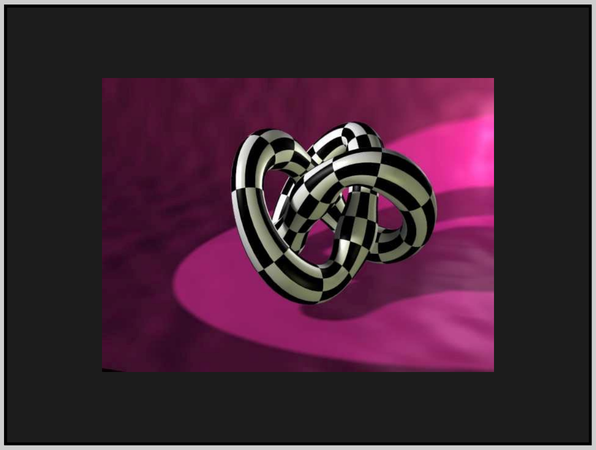


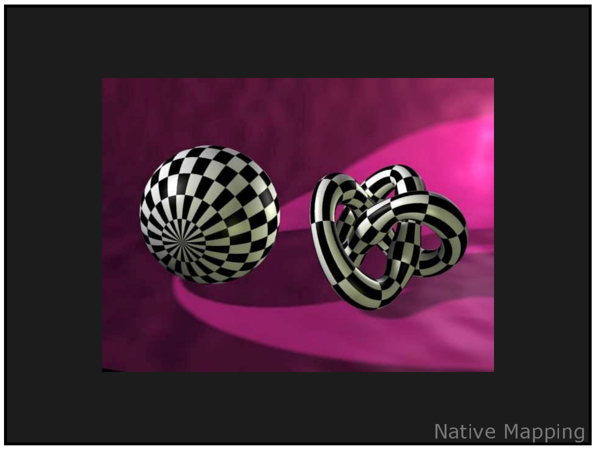


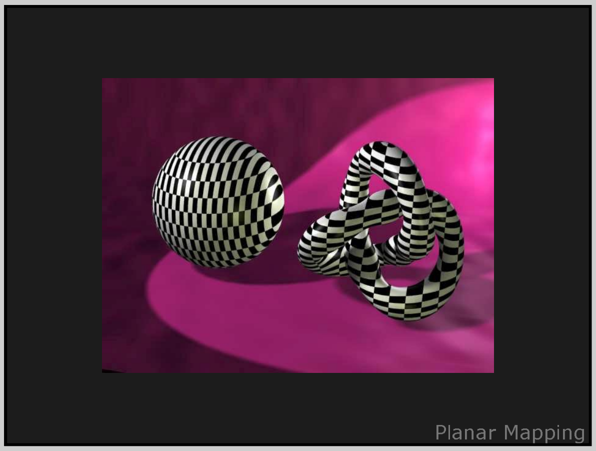


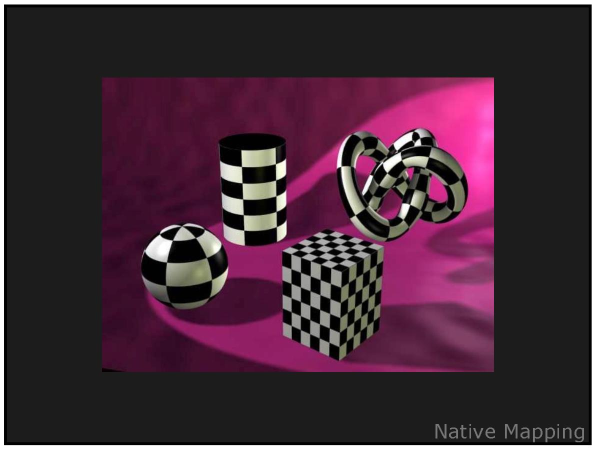


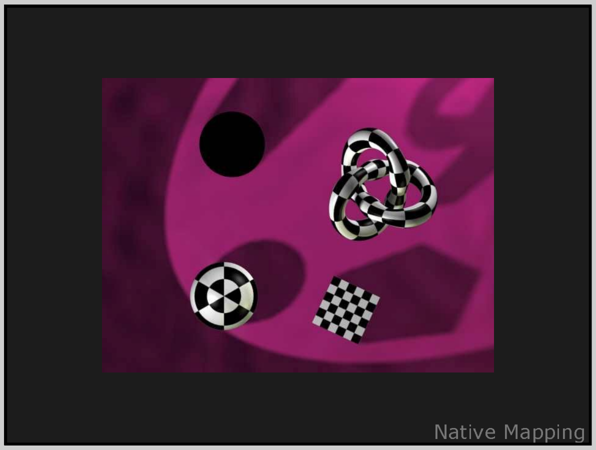
Texturing Methods

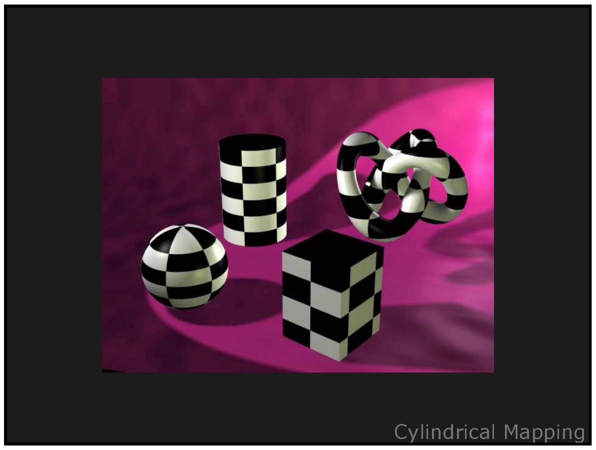


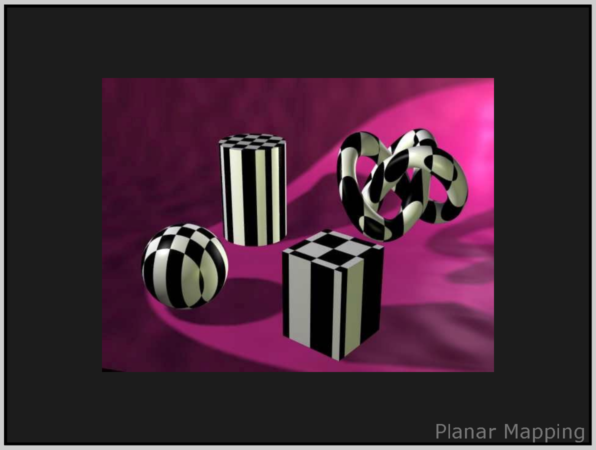


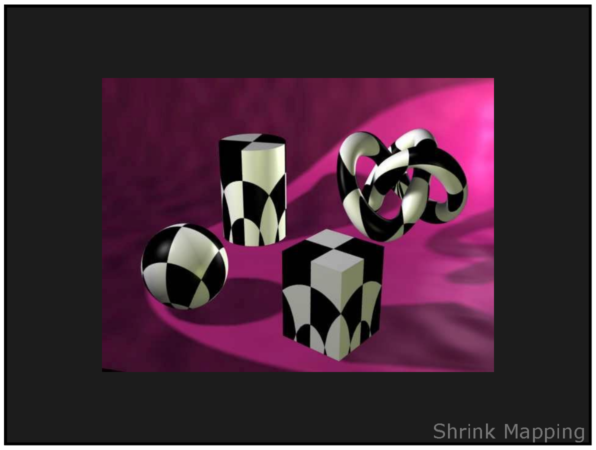


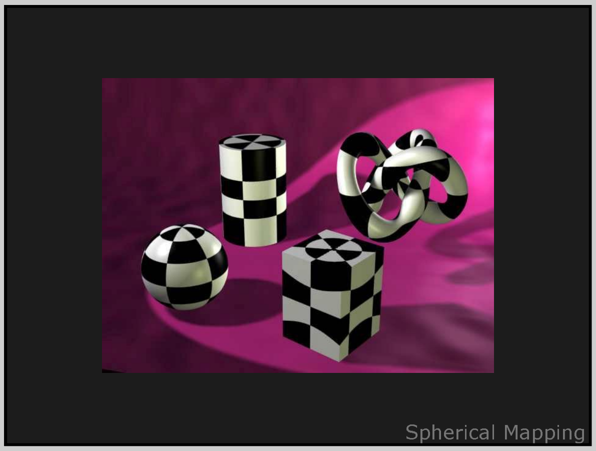


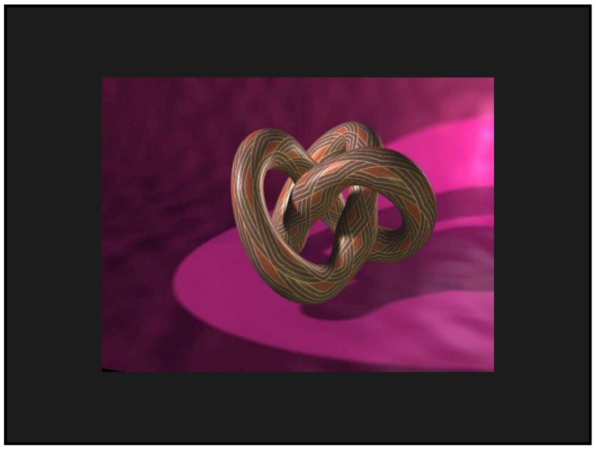

Native Mapping

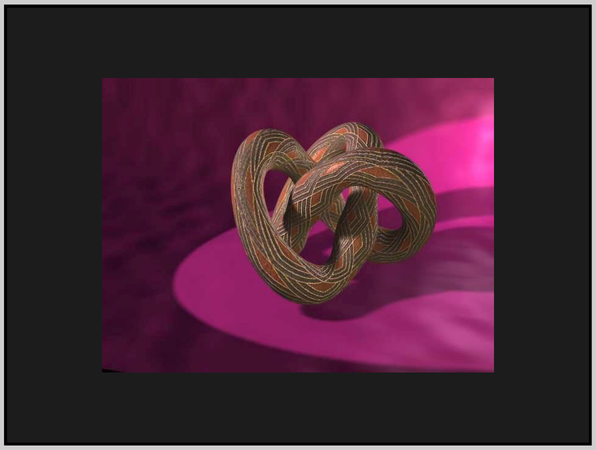

Cylindrical Mapping

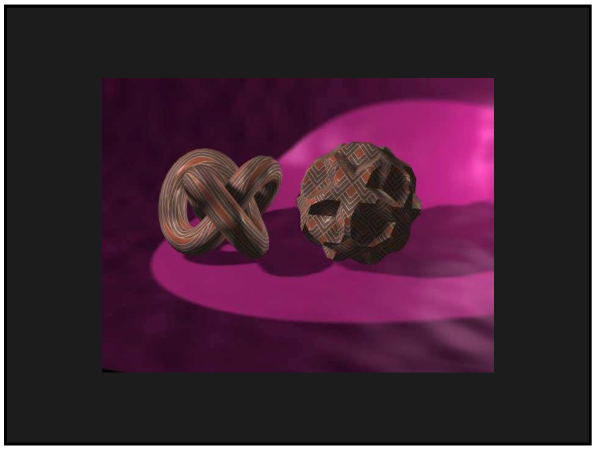

Planar Mapping

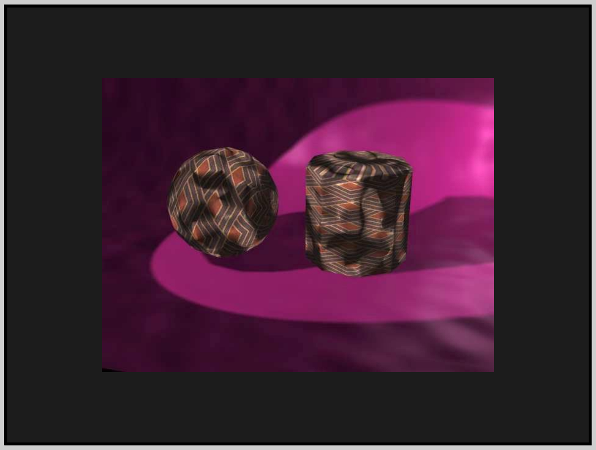

Shrink Mapping

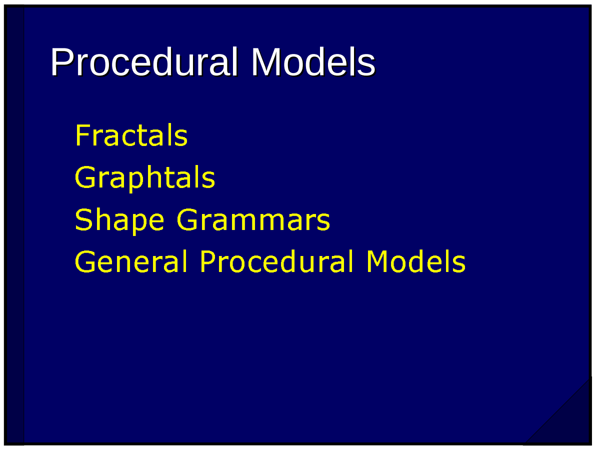

Spherical Mapping





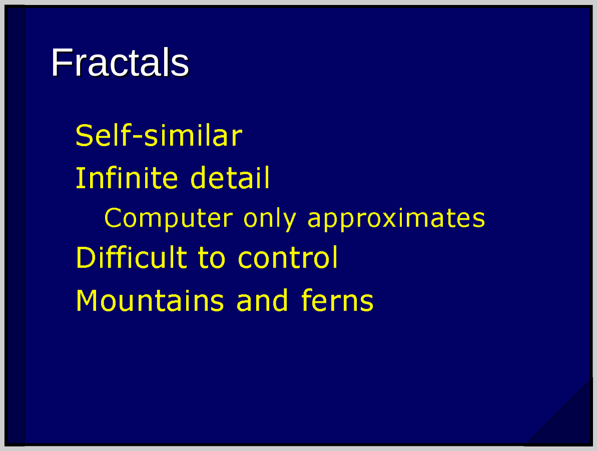






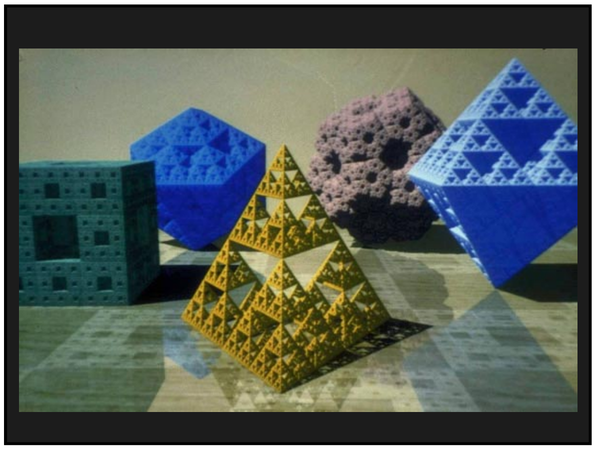
Procedural Models

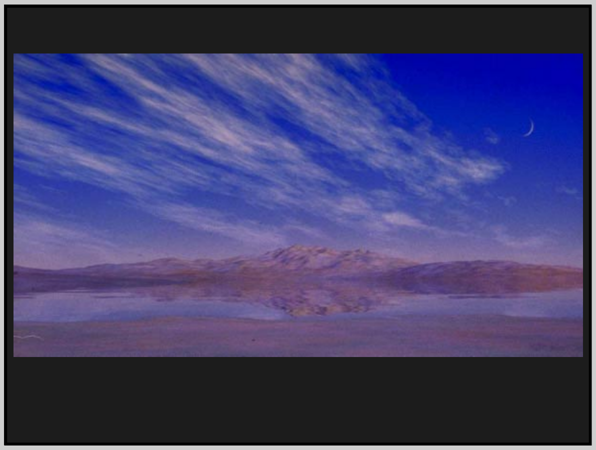
- Fractals
- Graphtals
- Shape Grammars
- General Procedural Models

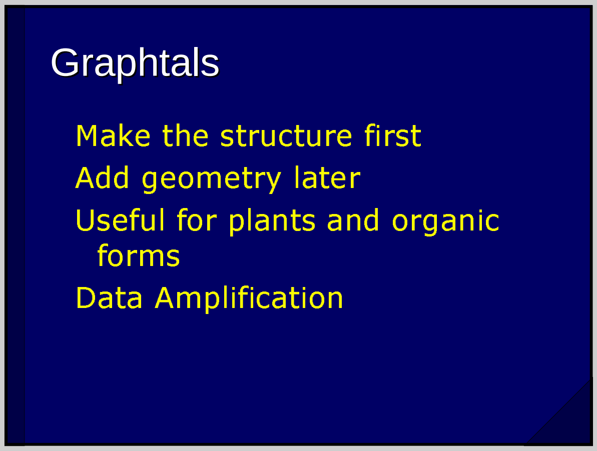


Fractals

- Self-similar
- Infinite detail
  - Computer only approximates
- Difficult to control
- Mountains and ferns

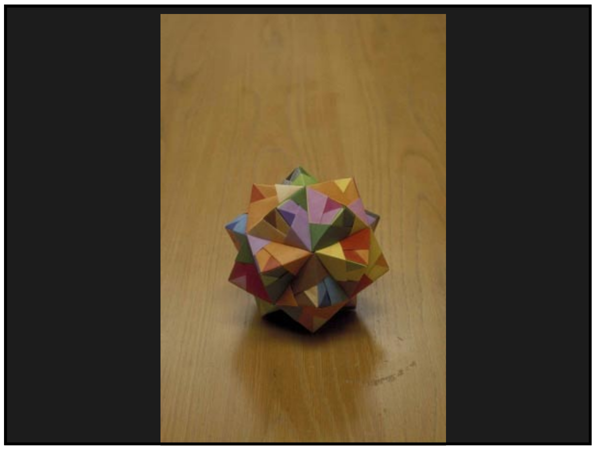


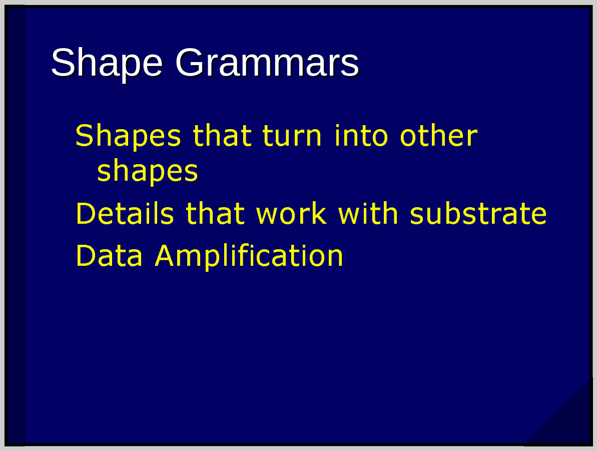




Graphtals

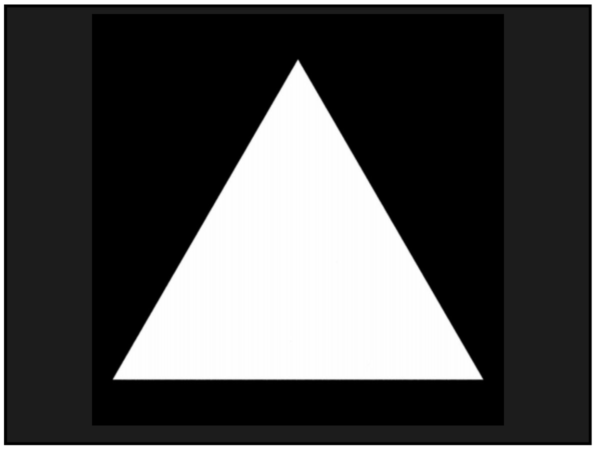
- Make the structure first
- Add geometry later
- Useful for plants and organic forms
- Data Amplification

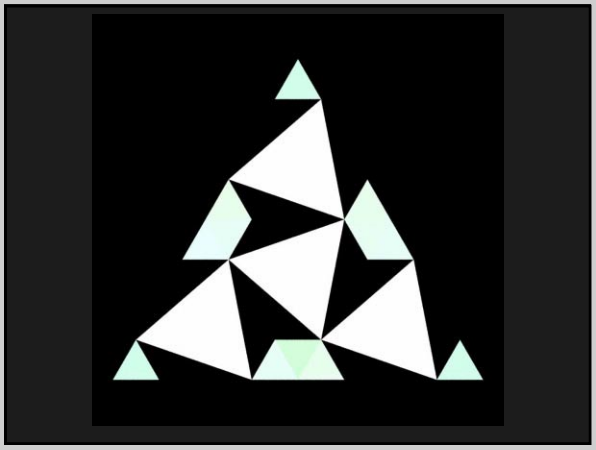


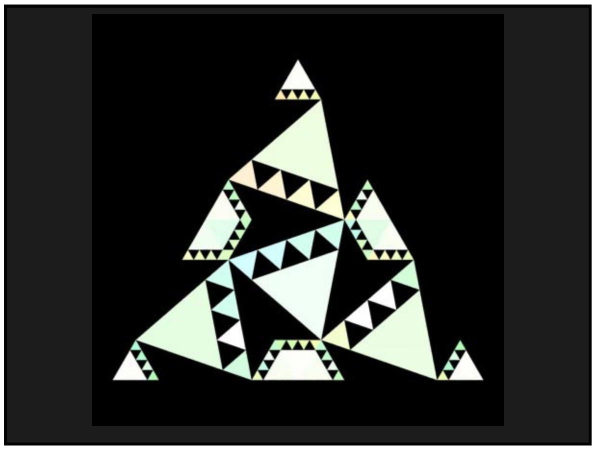


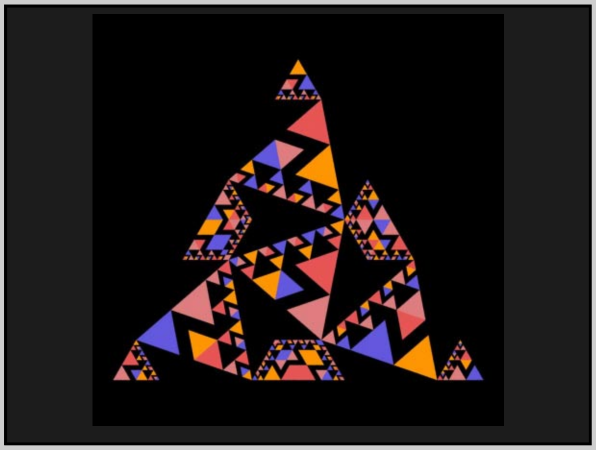
Shape Grammars

- Shapes that turn into other shapes
- Details that work with substrate
- Data Amplification

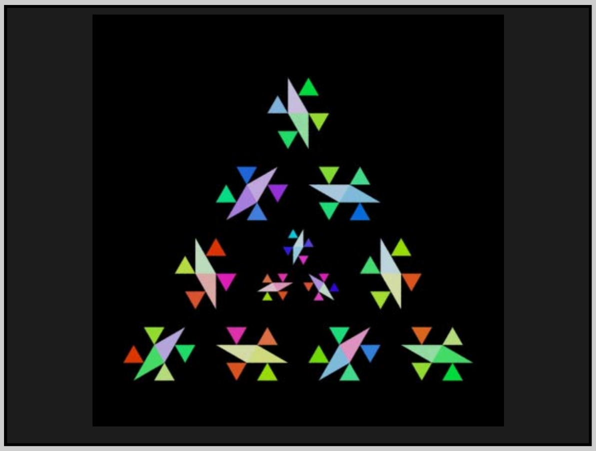


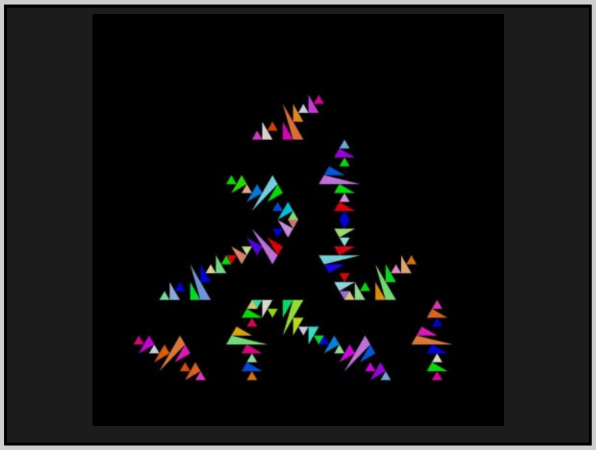


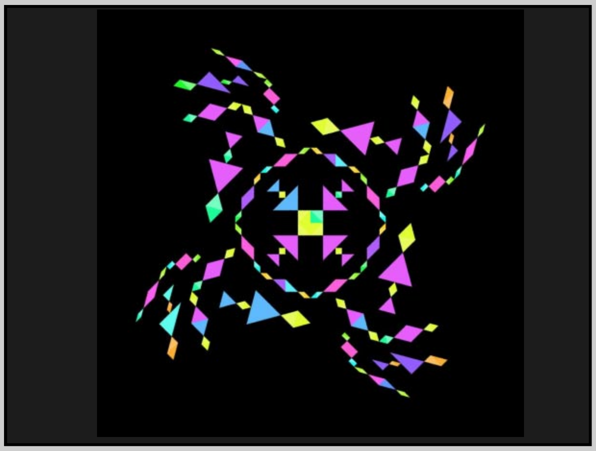


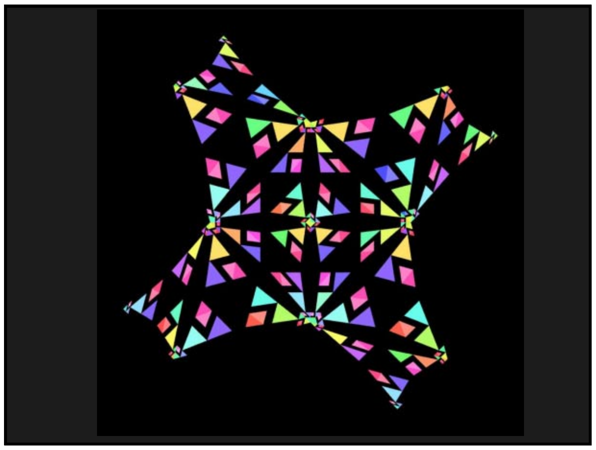


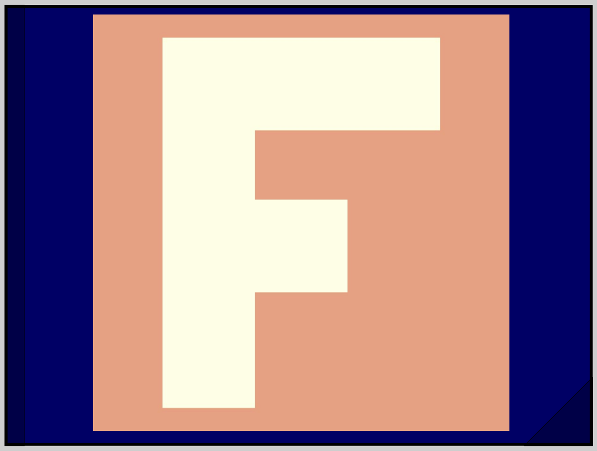
2D Variations

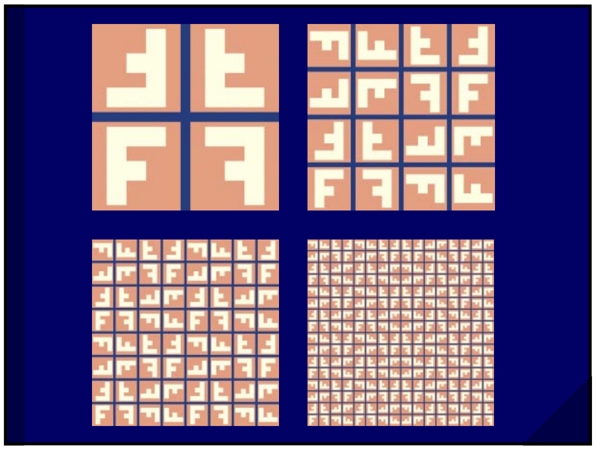


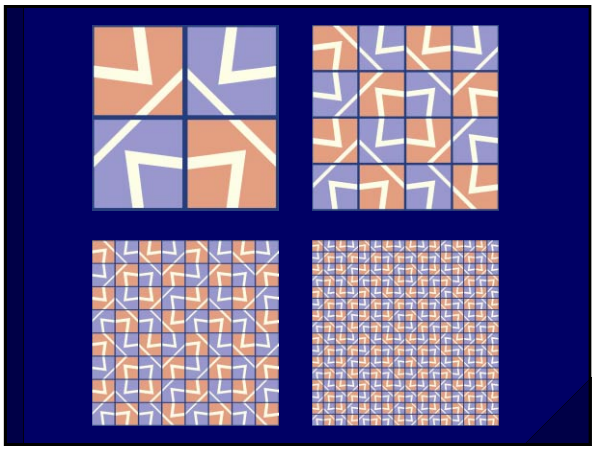


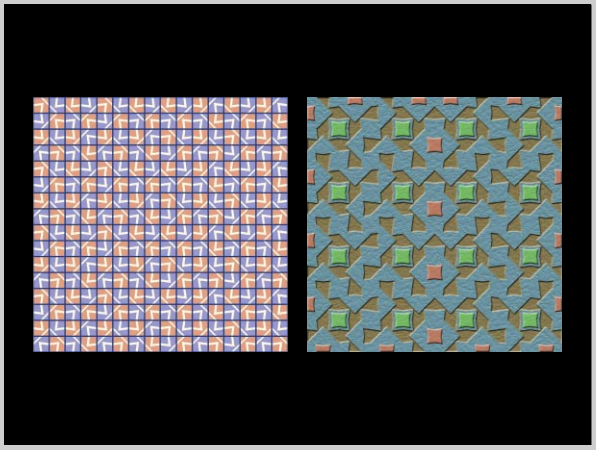


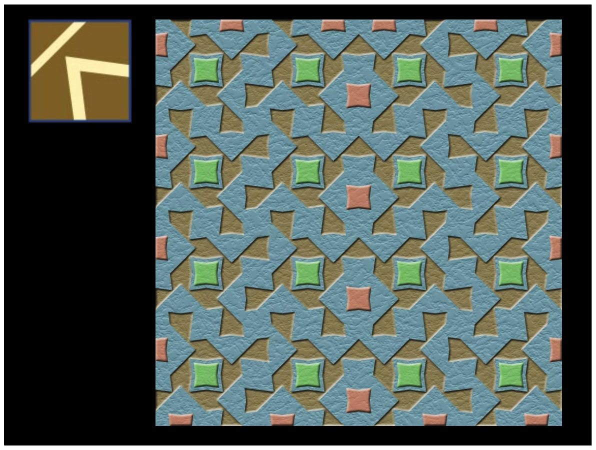


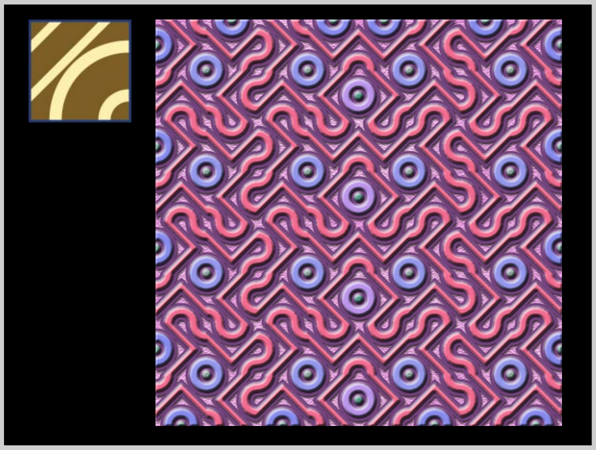


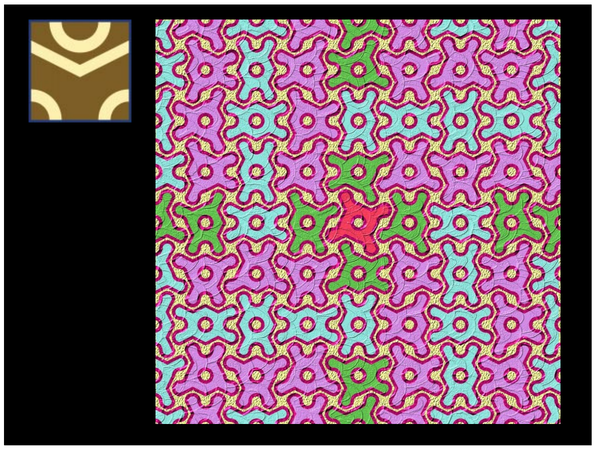


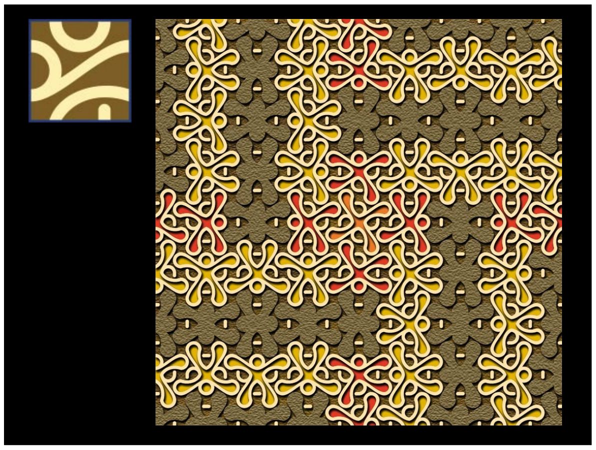


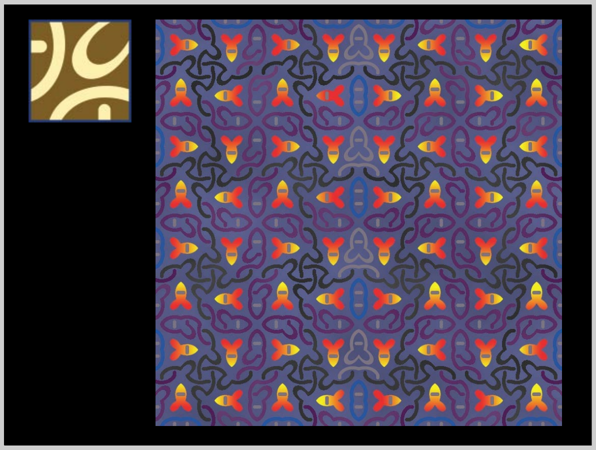


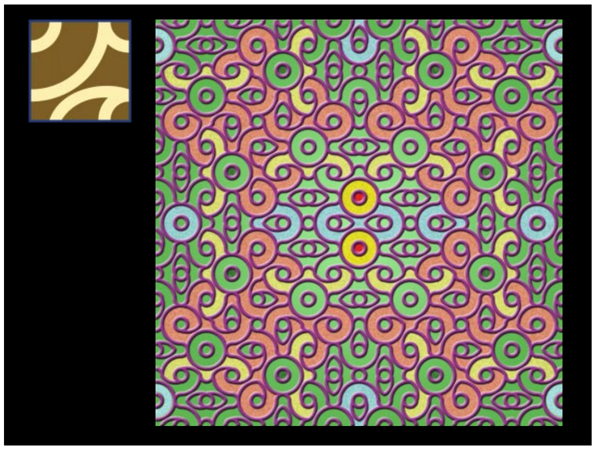


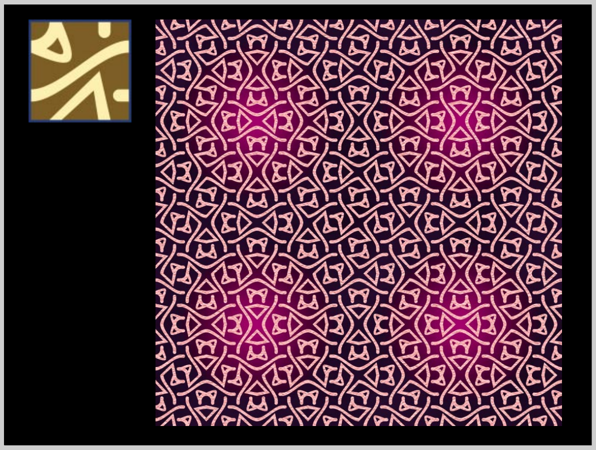


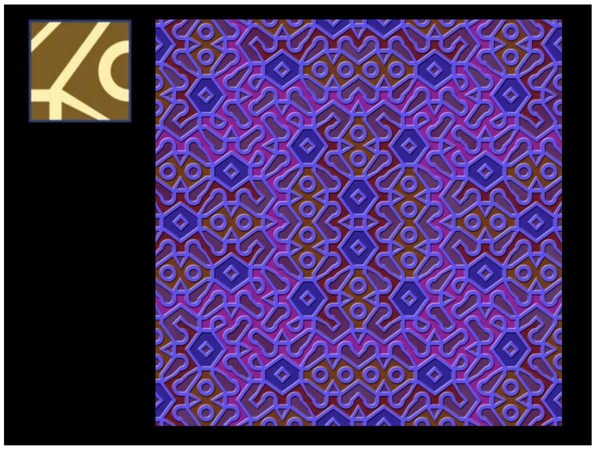


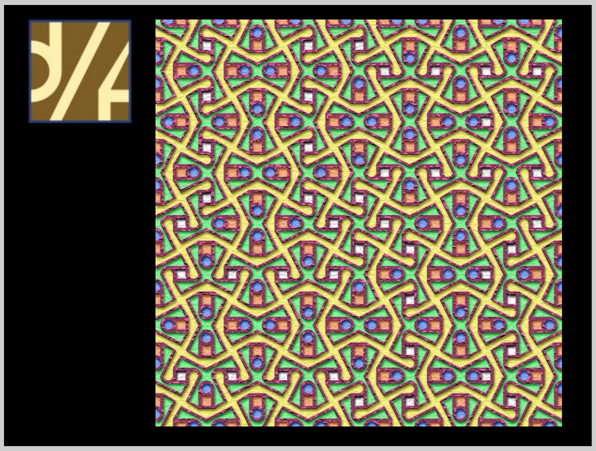


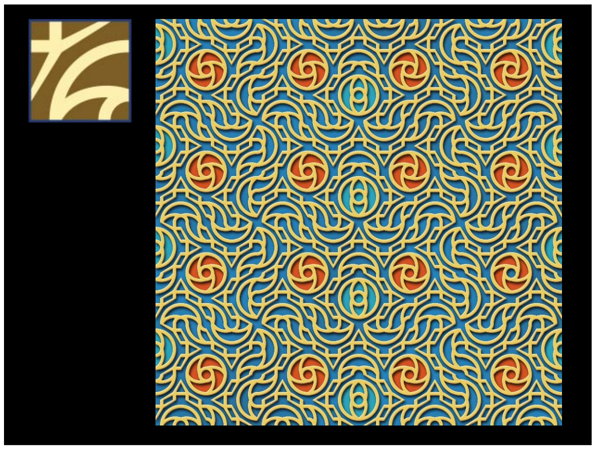


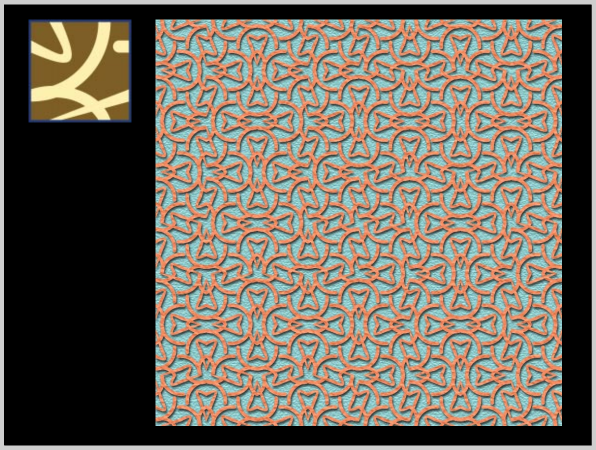


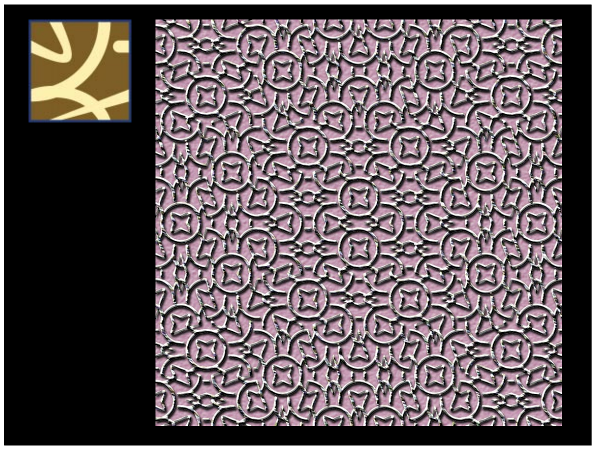


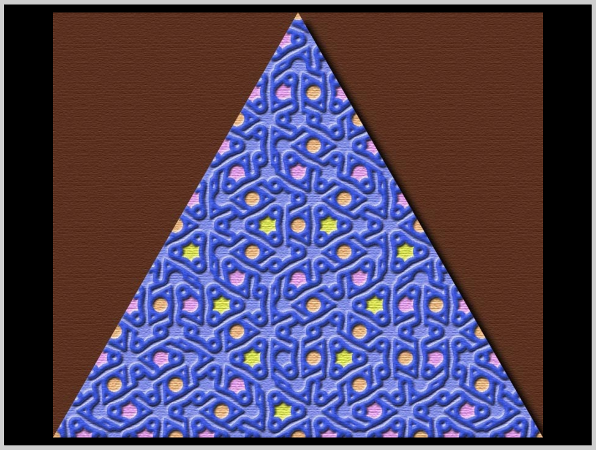


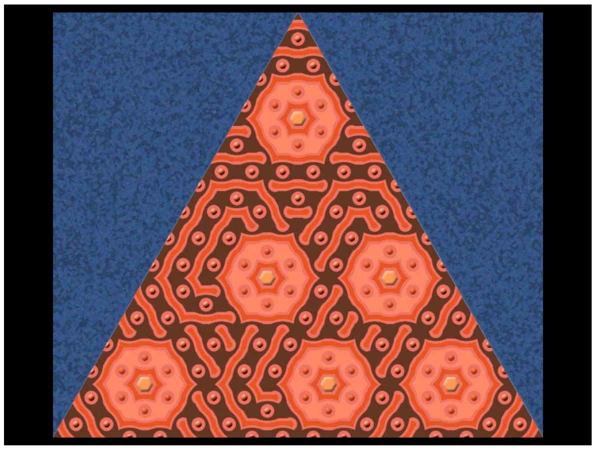


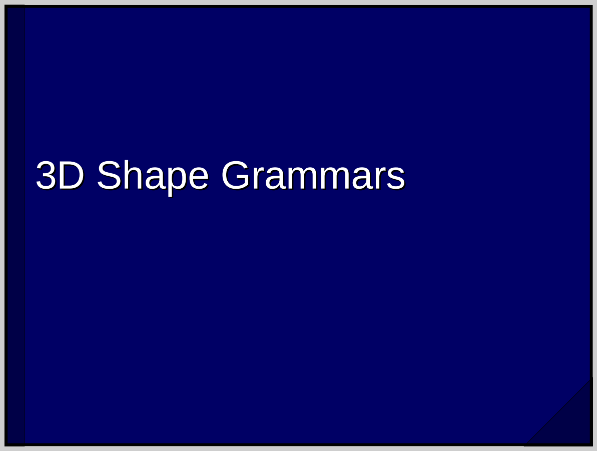




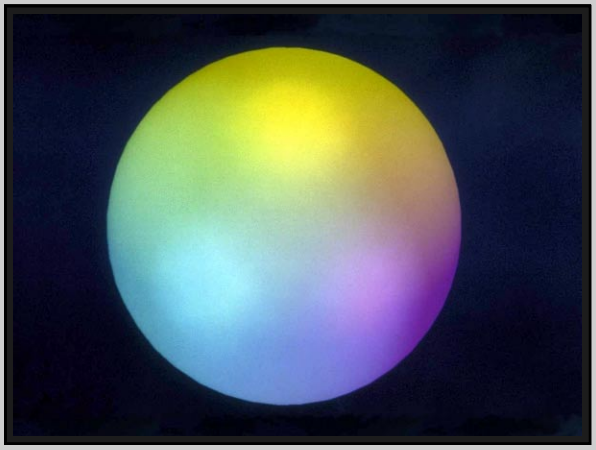


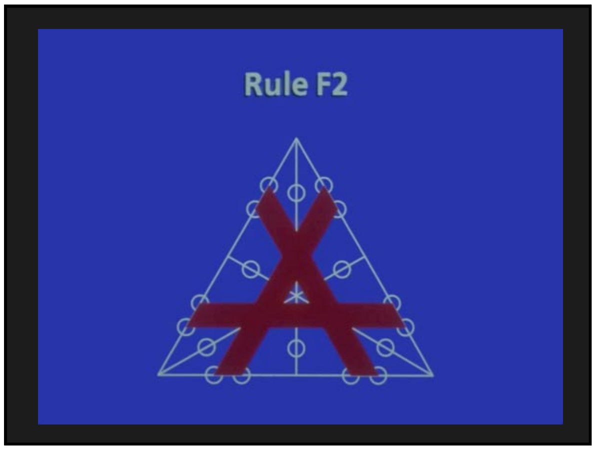




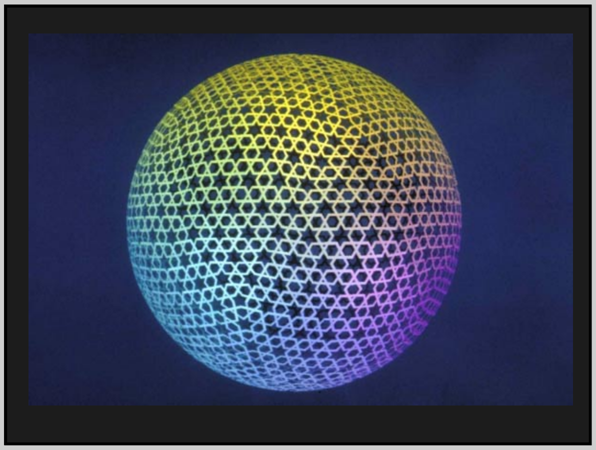


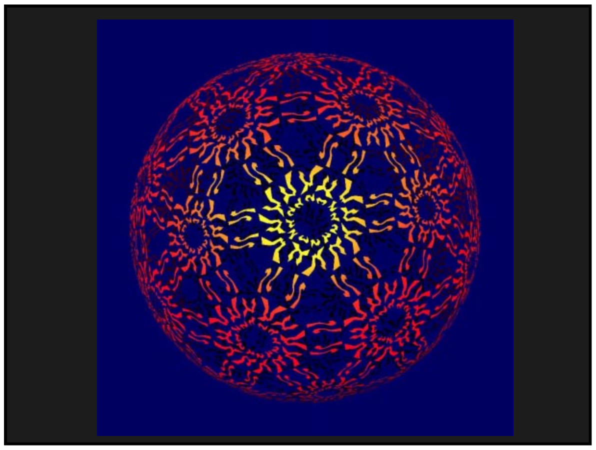
3D Shape Grammars

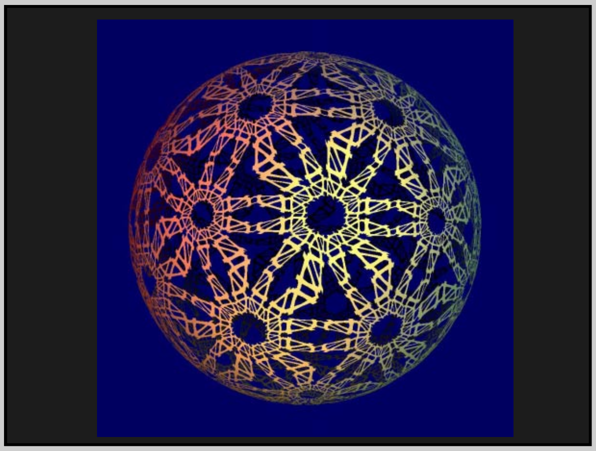


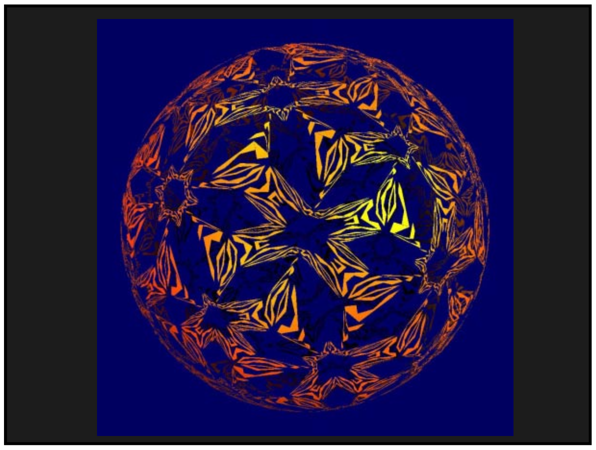


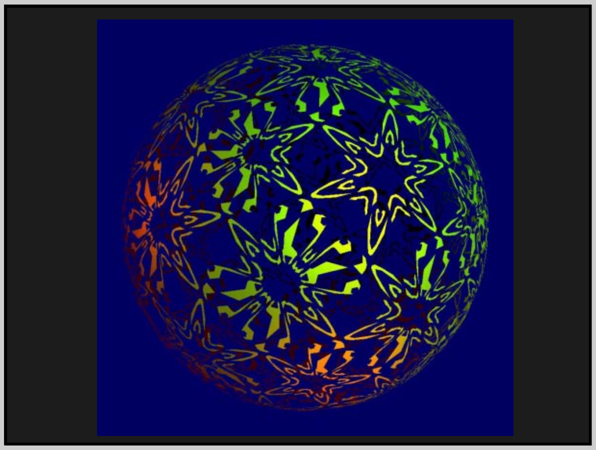
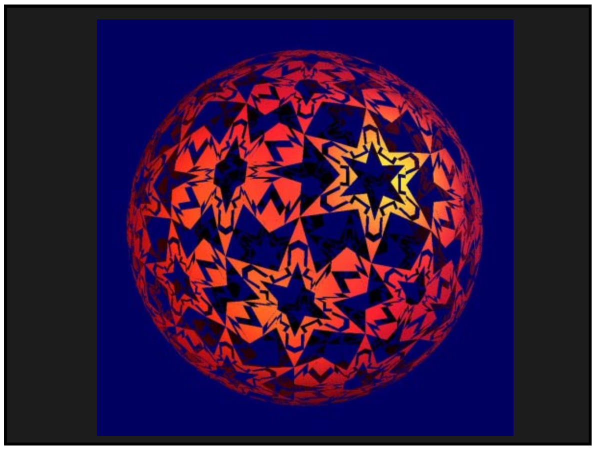
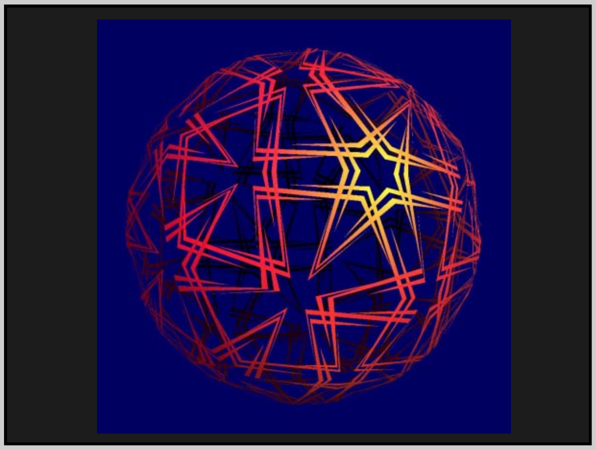
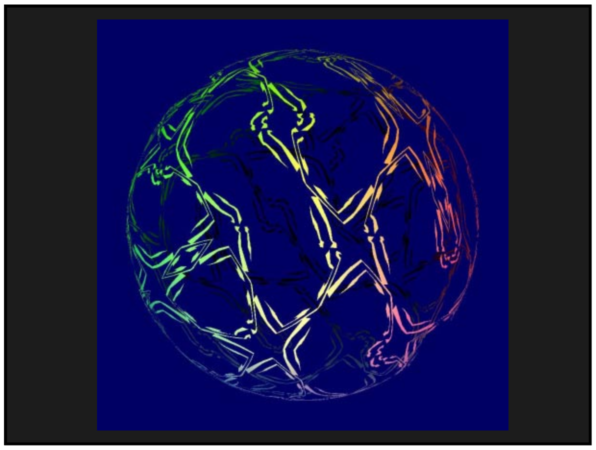
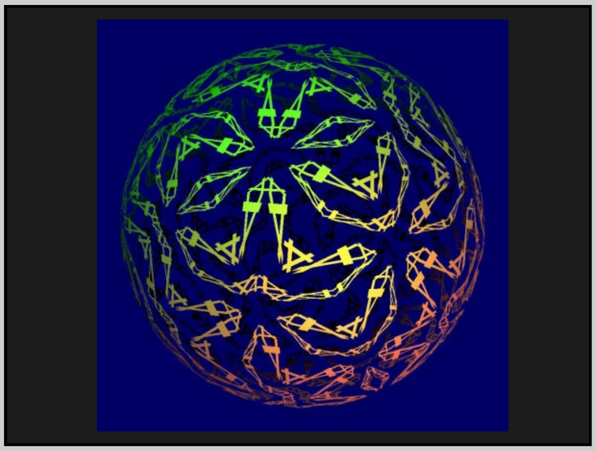
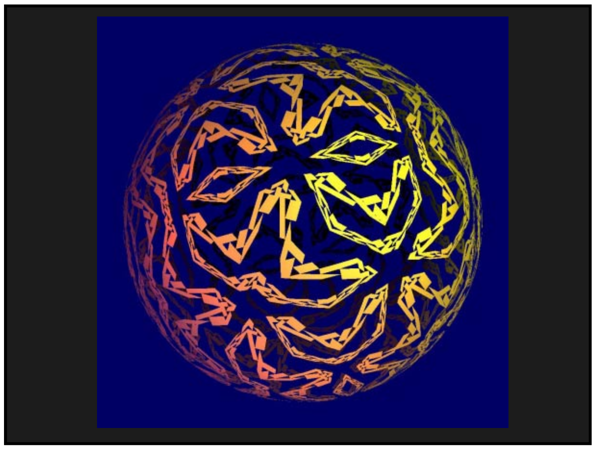
Rule F2

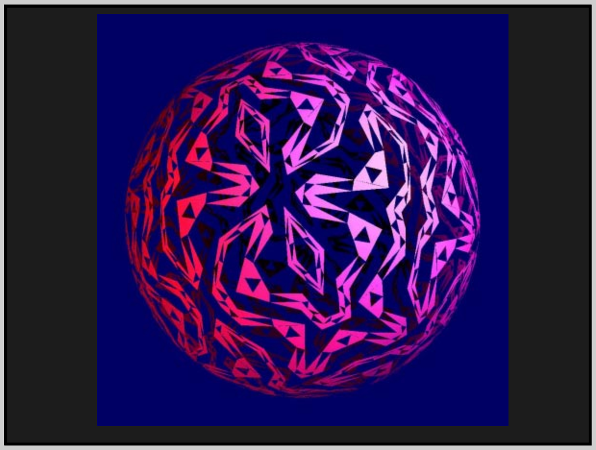
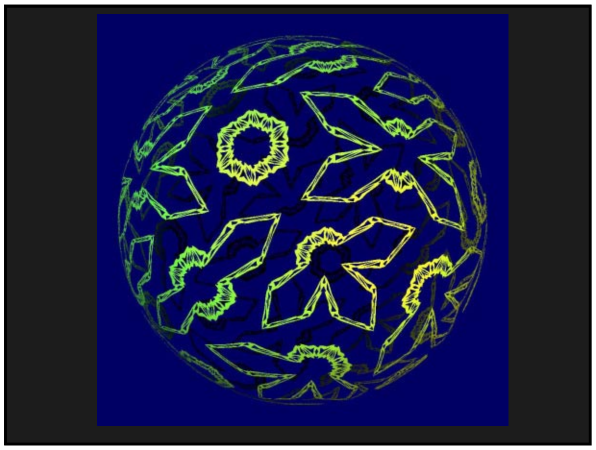
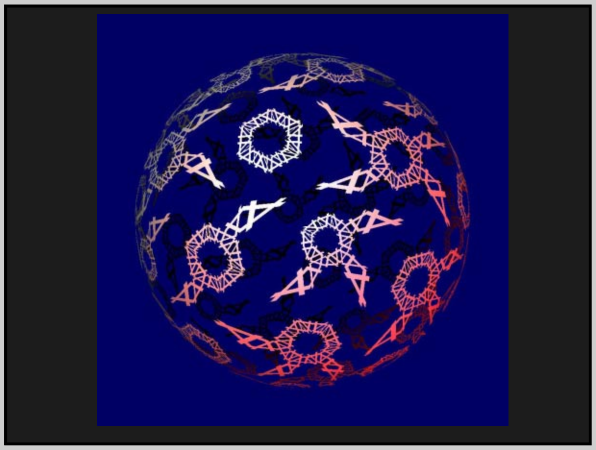
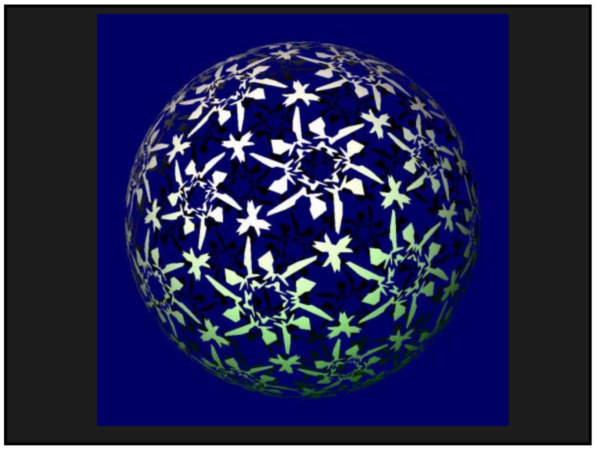
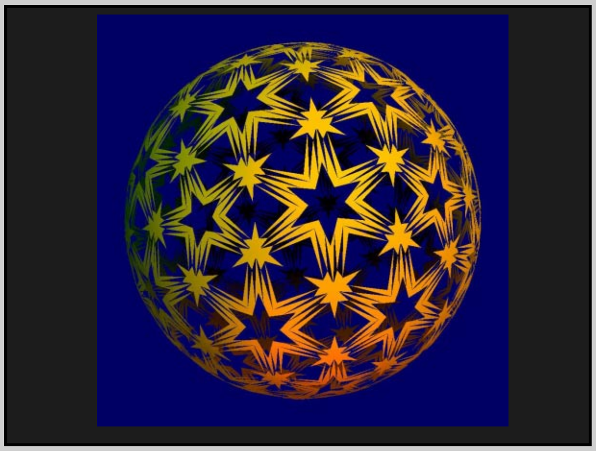


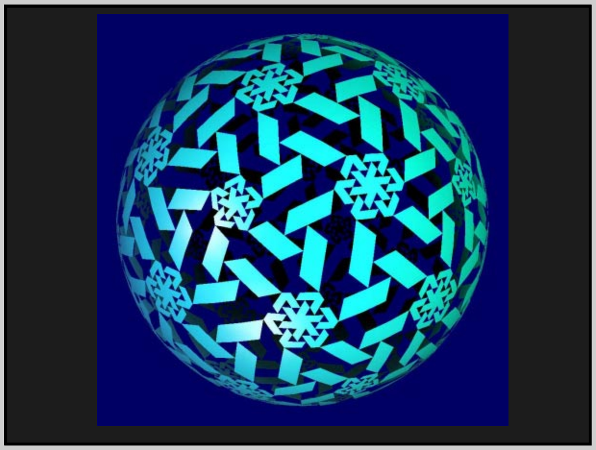


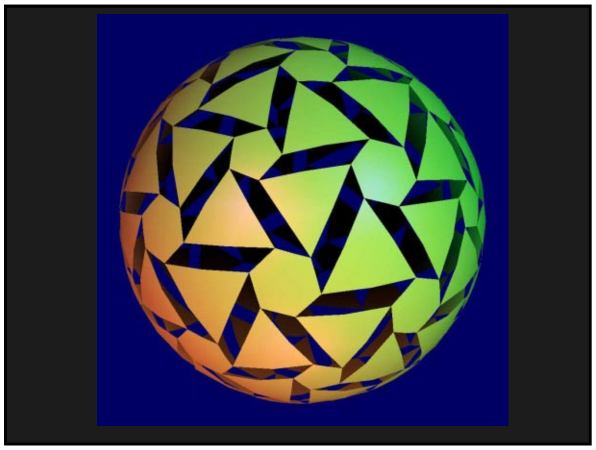


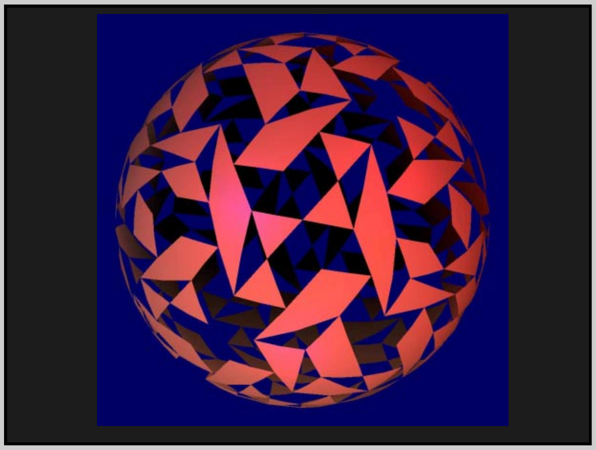


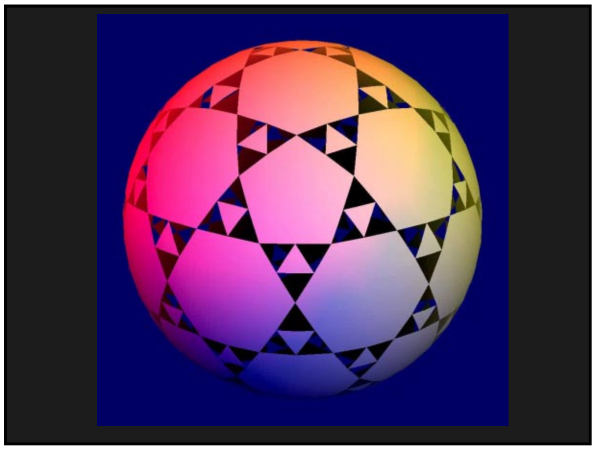







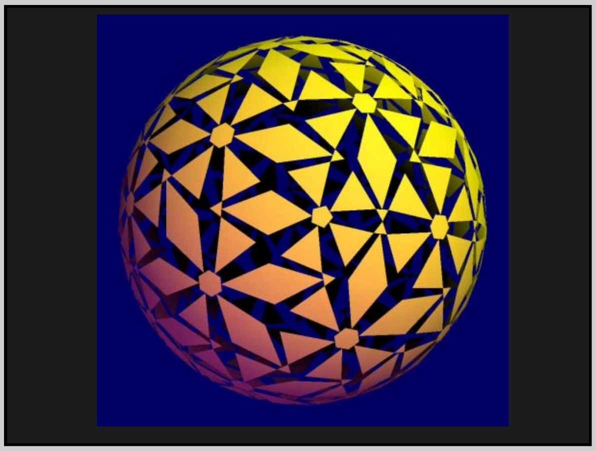






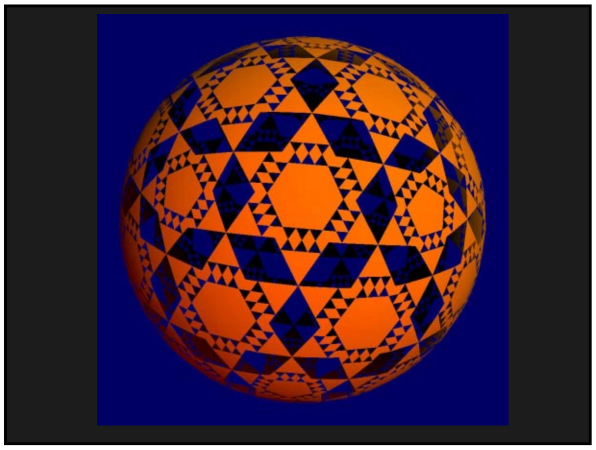


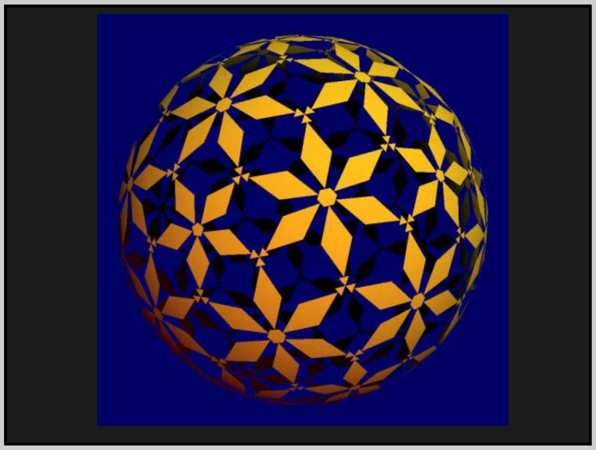


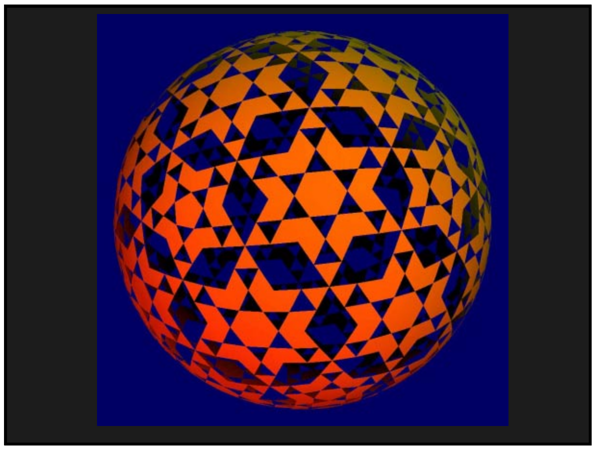


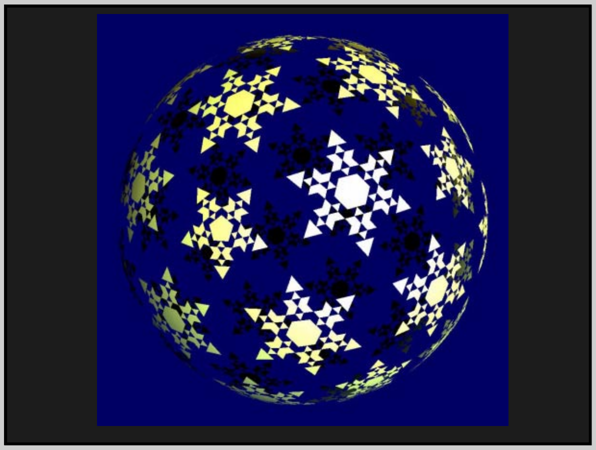


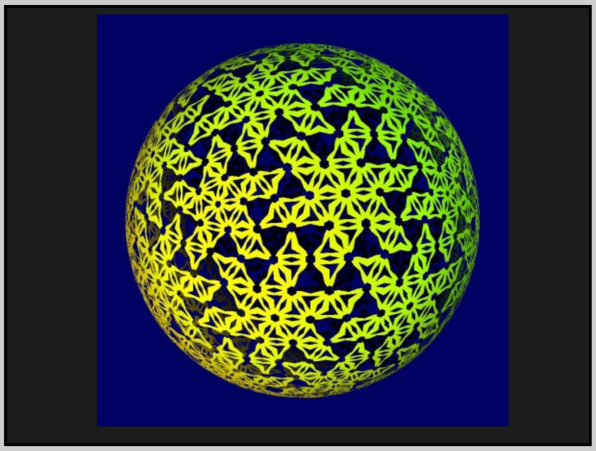


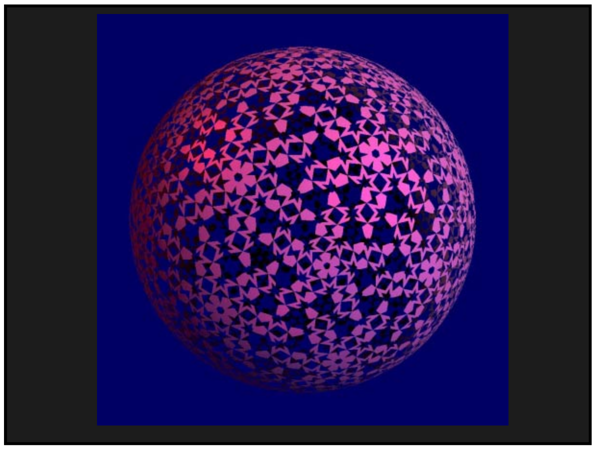


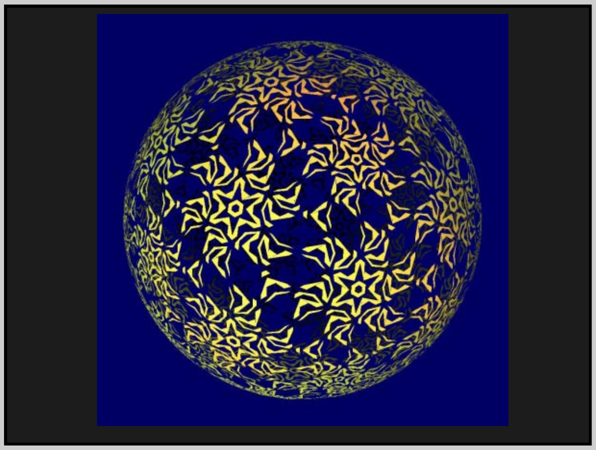


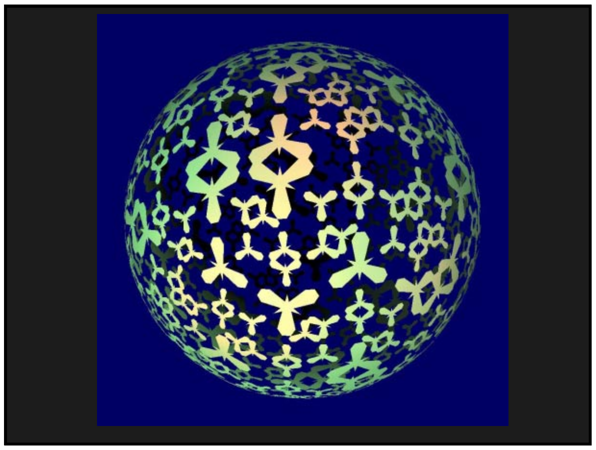


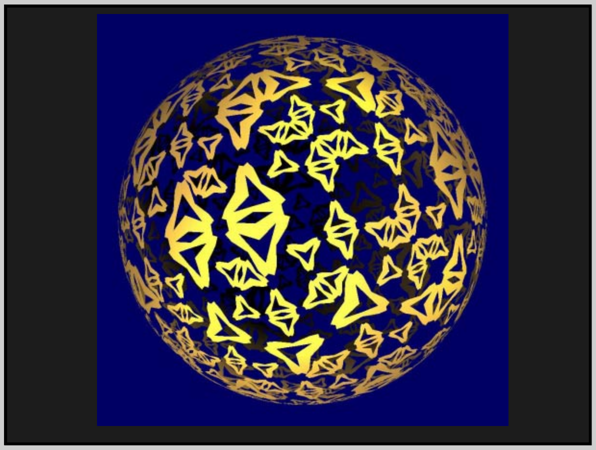


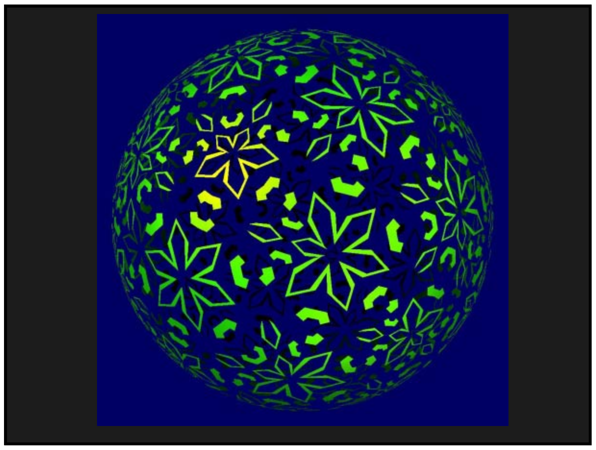


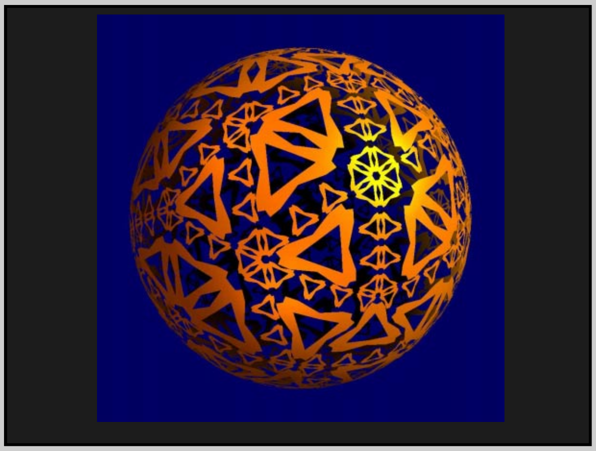


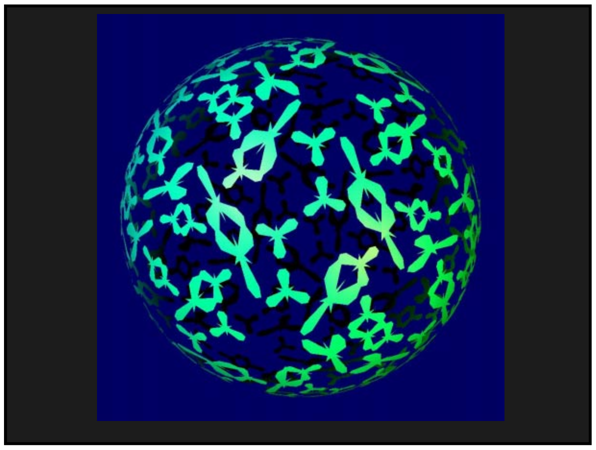


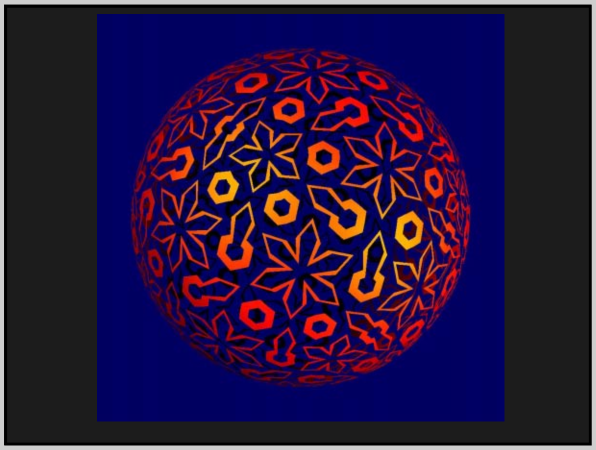


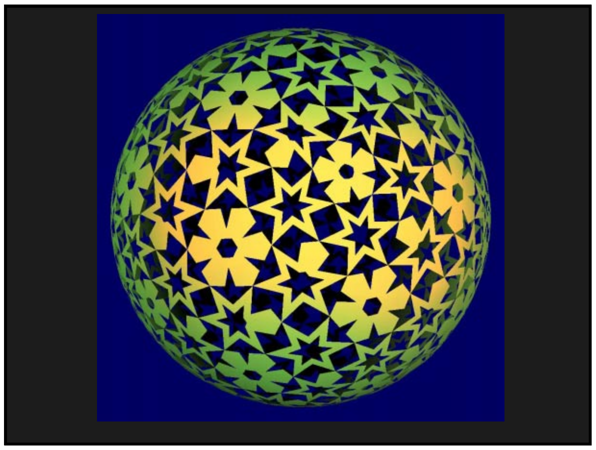


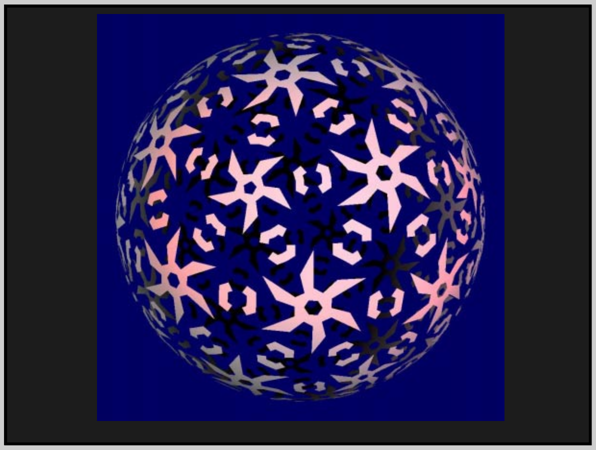


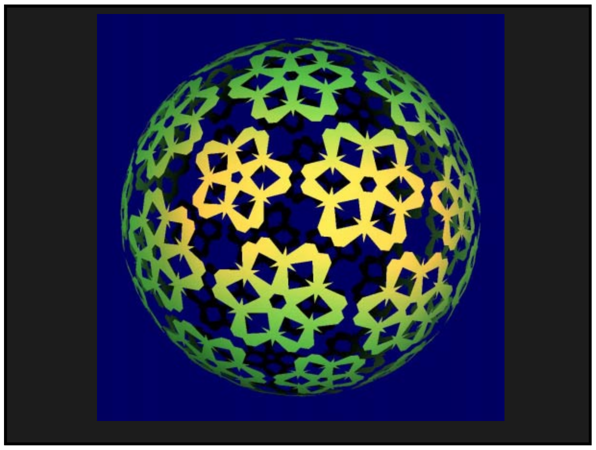


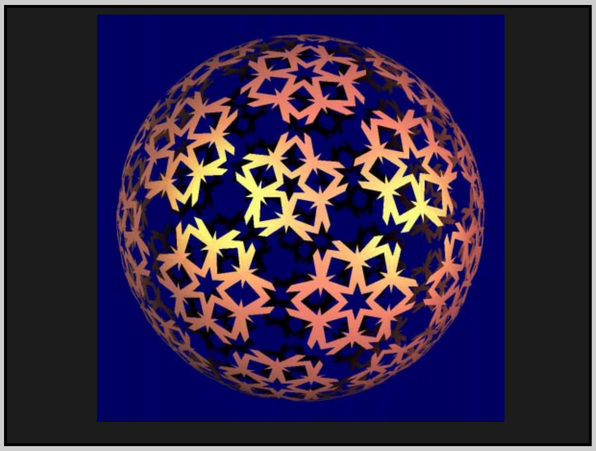


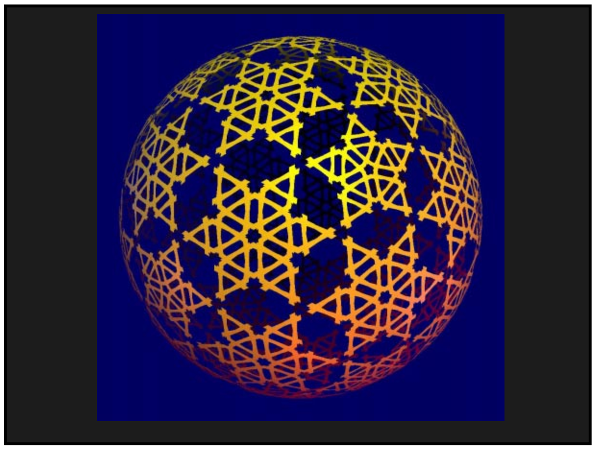


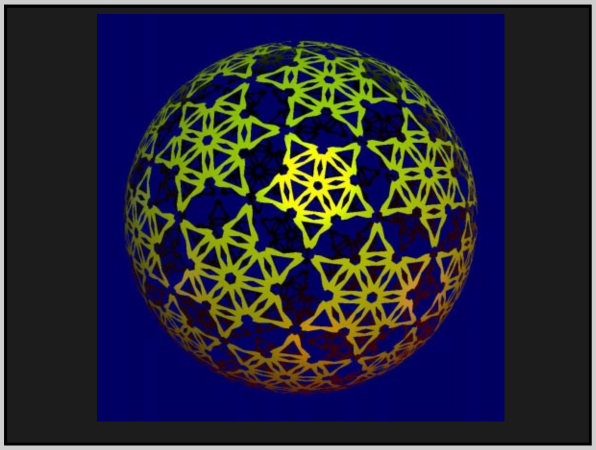


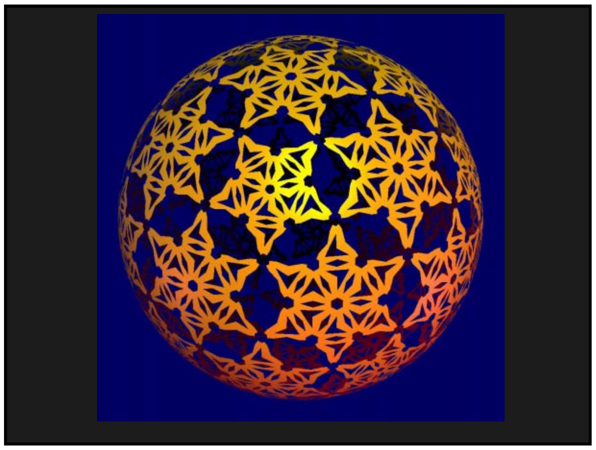


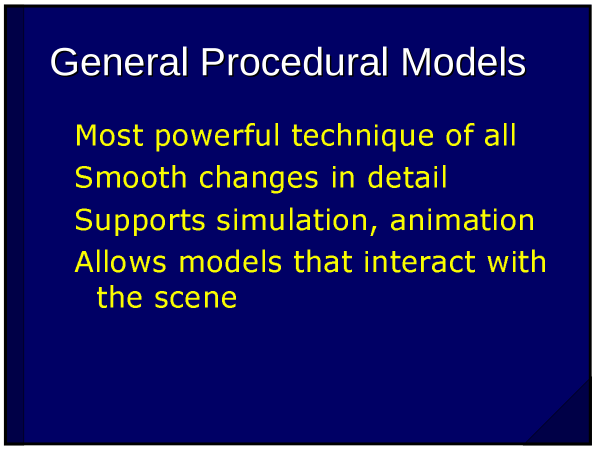






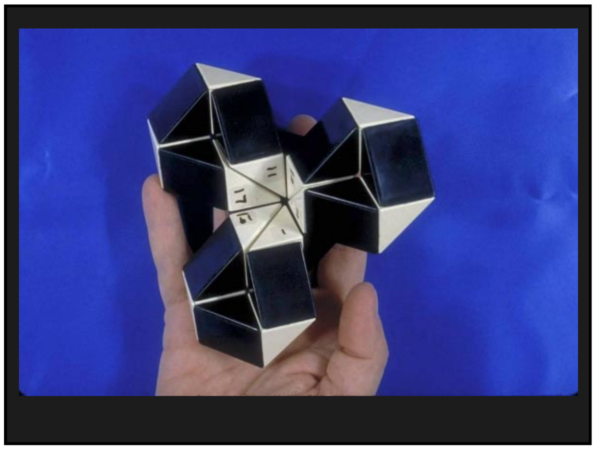


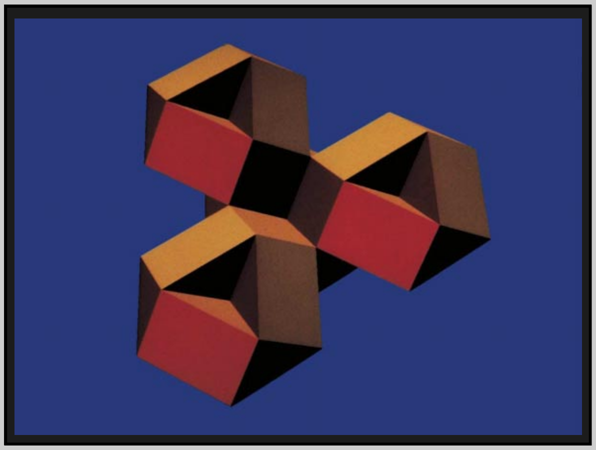


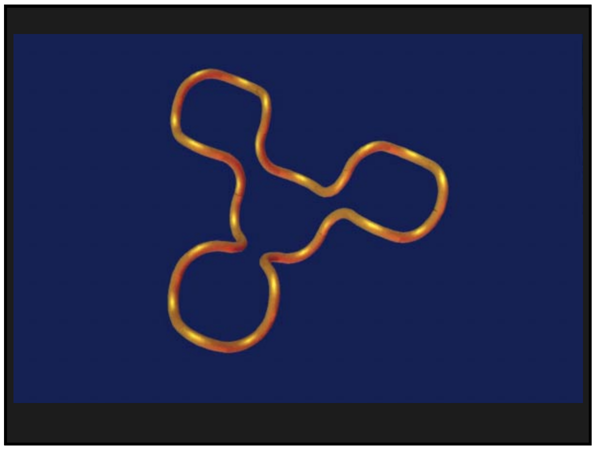


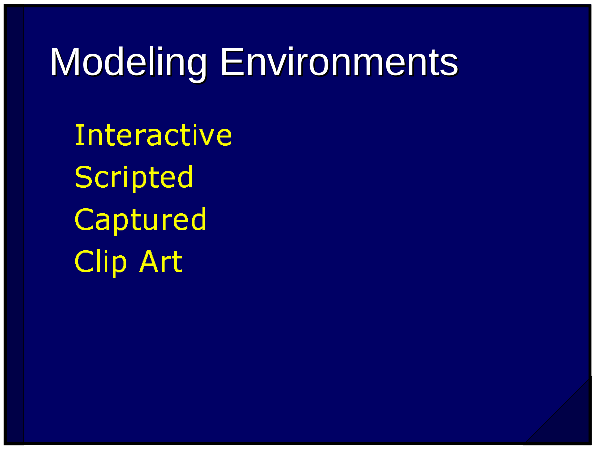
General Procedural Models

- Most powerful technique of all
- Smooth changes in detail
- Supports simulation, animation
- Allows models that interact with the scene



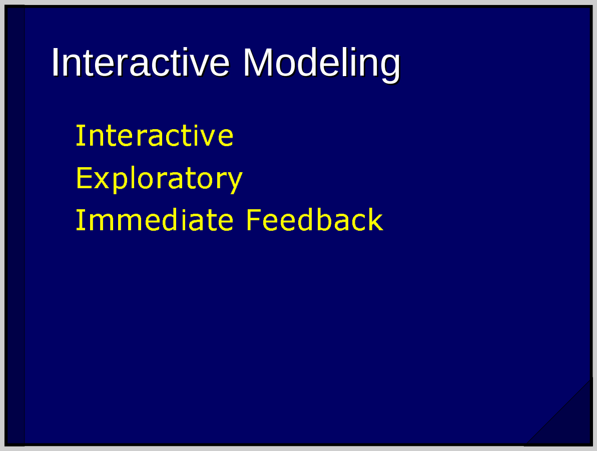






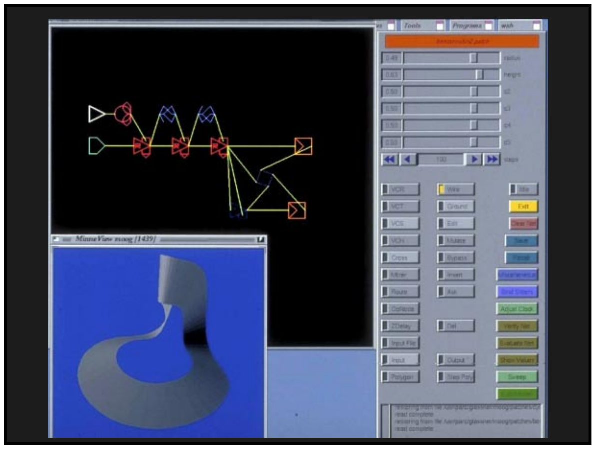
Modeling Environments

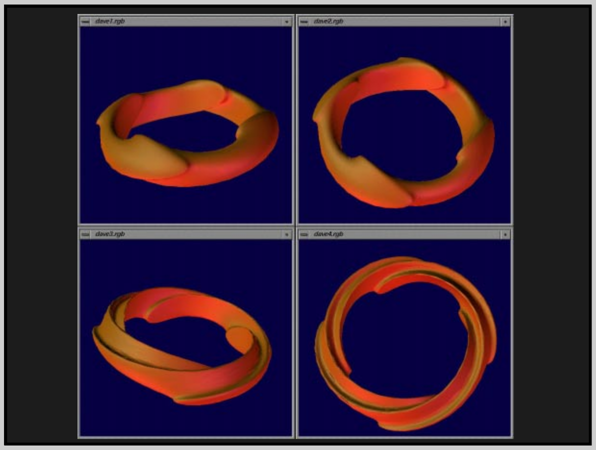
- Interactive
- Scripted
- Captured
- Clip Art

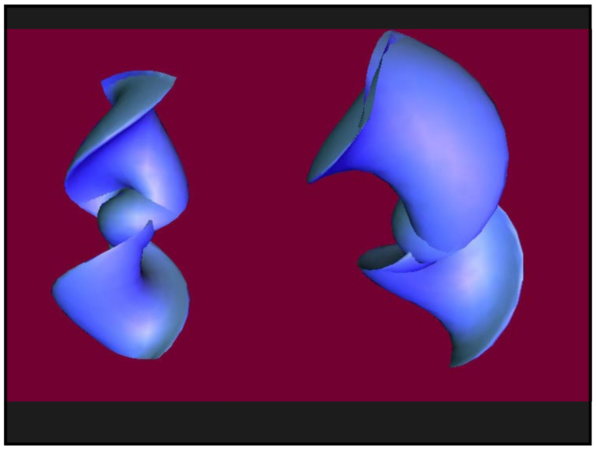


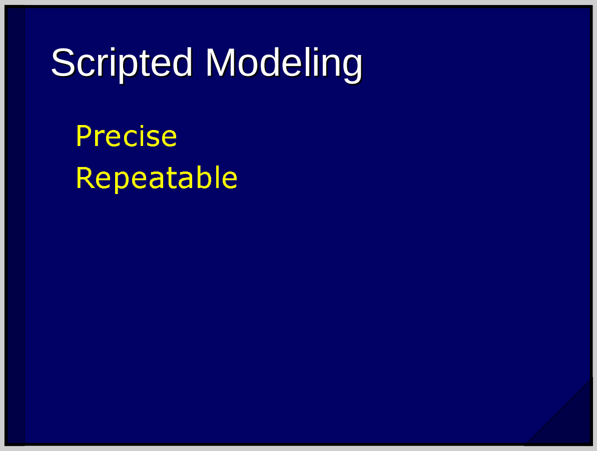
Interactive Modeling

- Interactive
- Exploratory
- Immediate Feedback



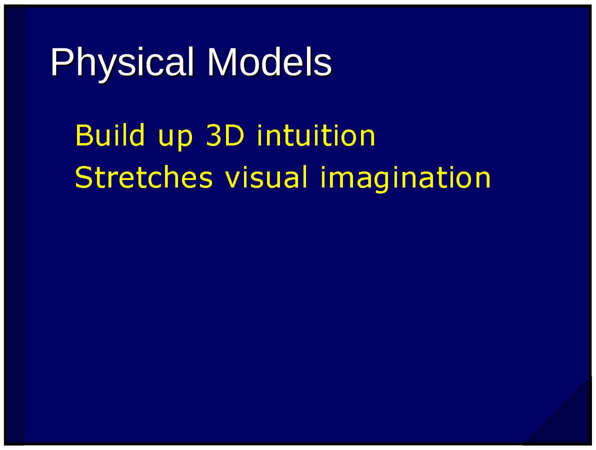






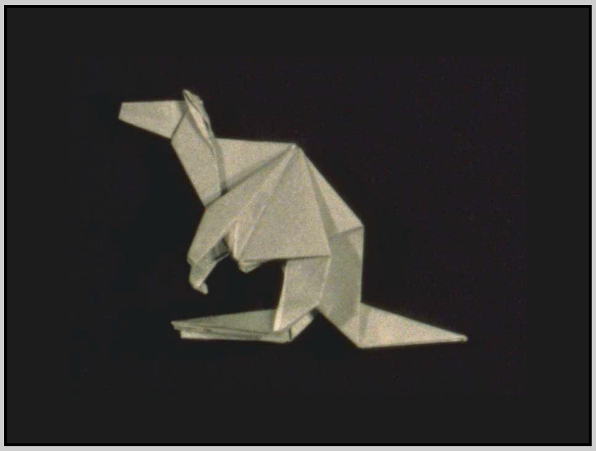
Scripted Modeling

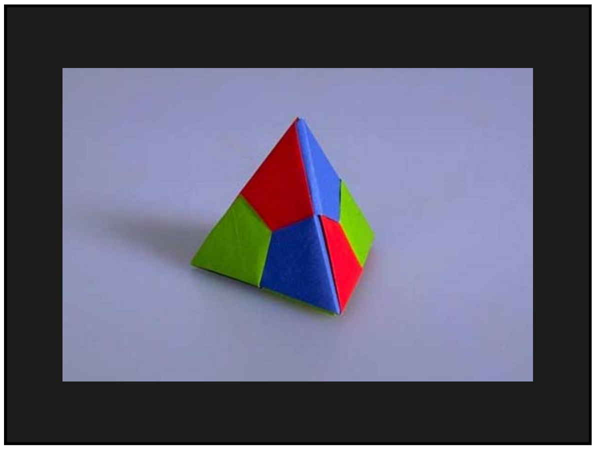
- Precise
- Repeatable

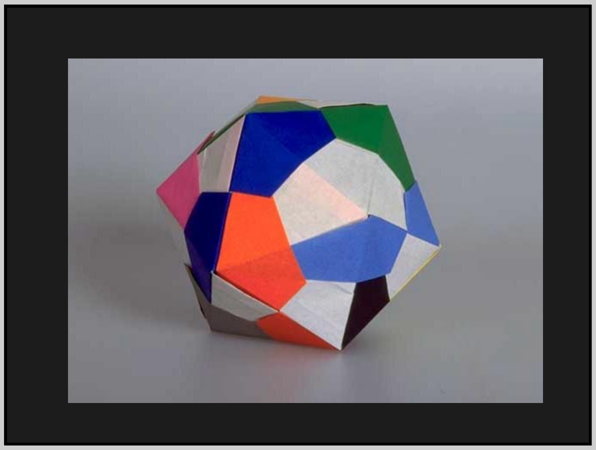
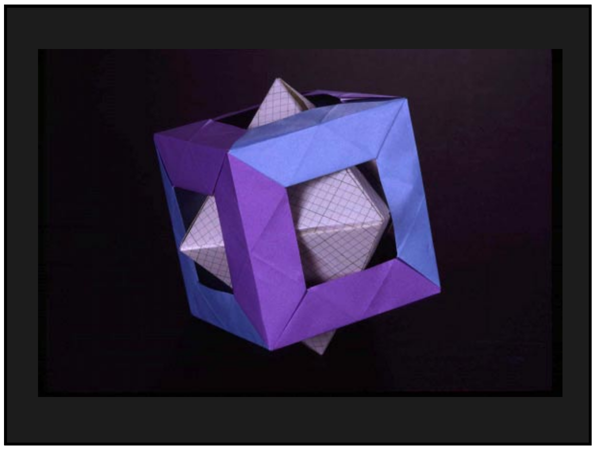


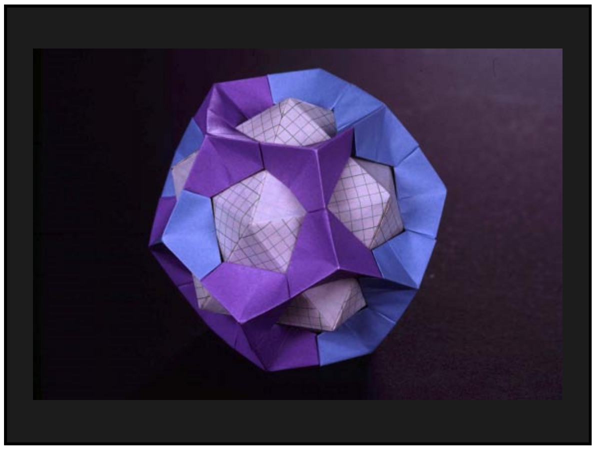
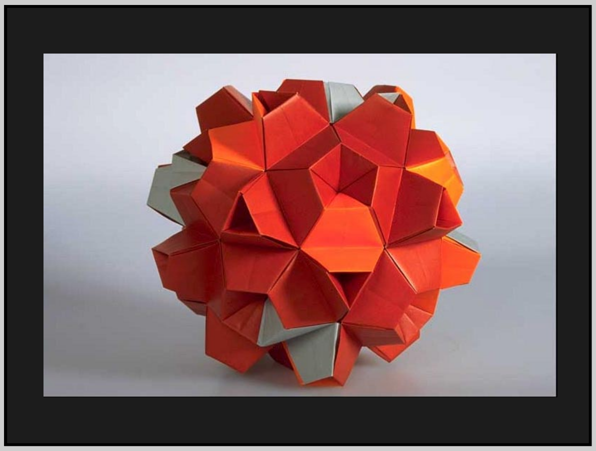
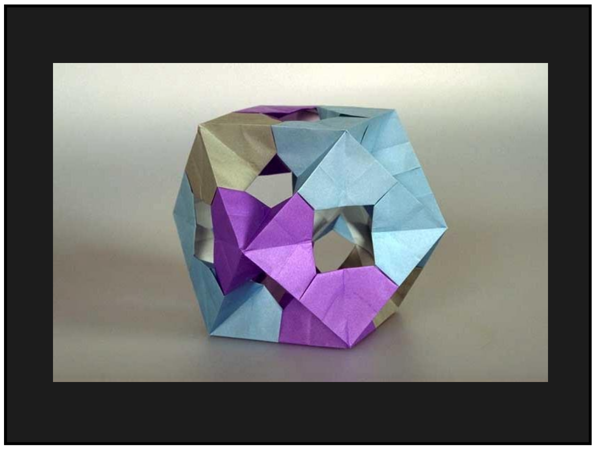
Physical Models

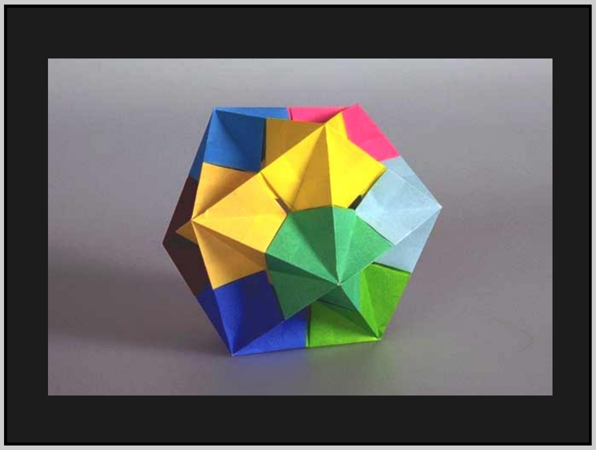
- Build up 3D intuition
- Stretches visual imagination

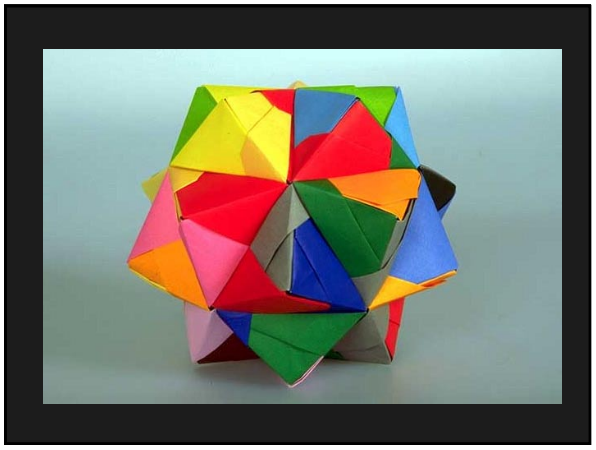


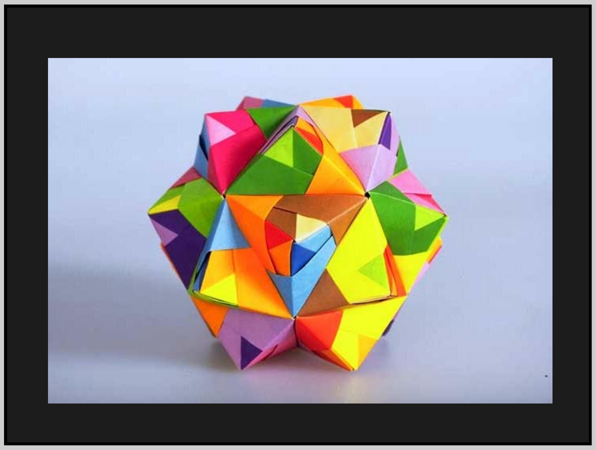


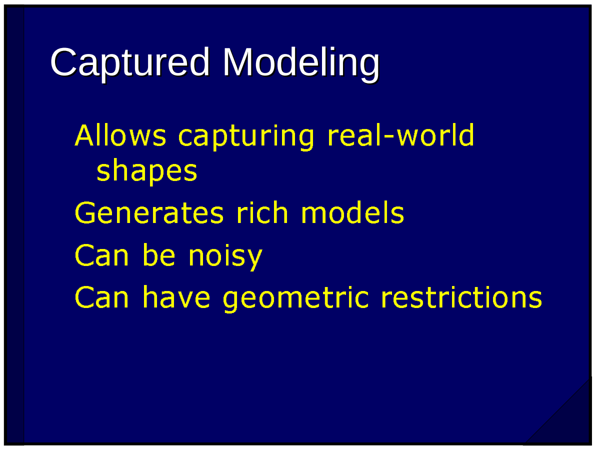





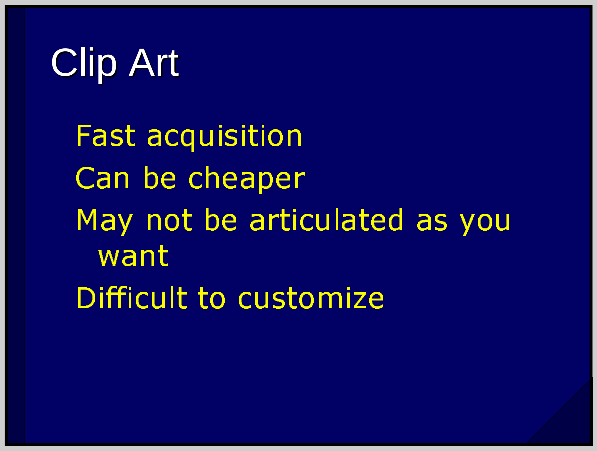






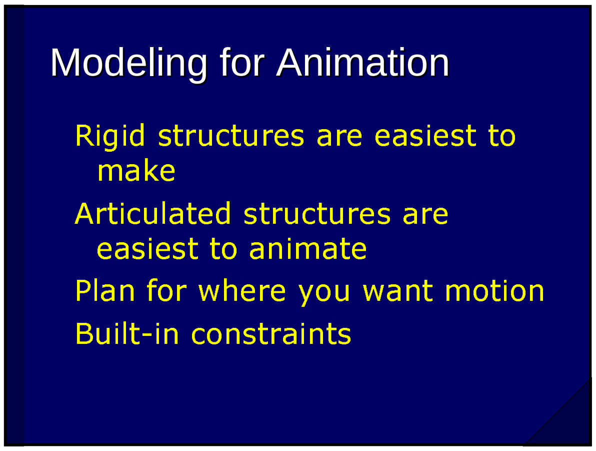
Captured Modeling

- Allows capturing real-world shapes
- Generates rich models
- Can be noisy
- Can have geometric restrictions



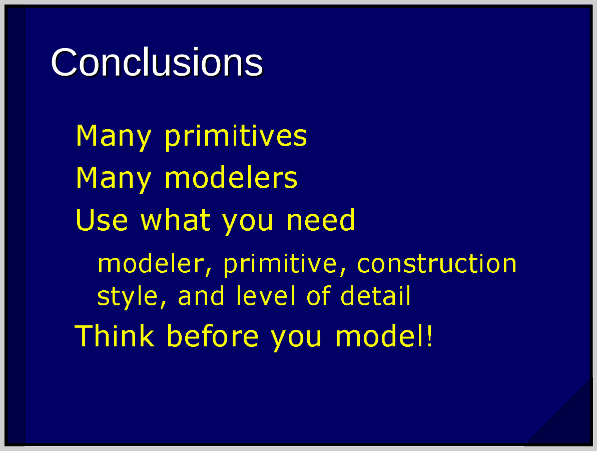
Clip Art

- Fast acquisition
- Can be cheaper
- May not be articulated as you want
- Difficult to customize



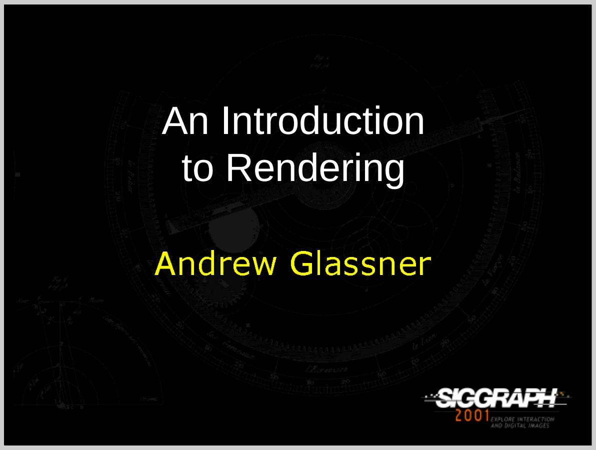
Modeling for Animation

- Rigid structures are easiest to make
- Articulated structures are easiest to animate
- Plan for where you want motion
- Built-in constraints



Conclusions

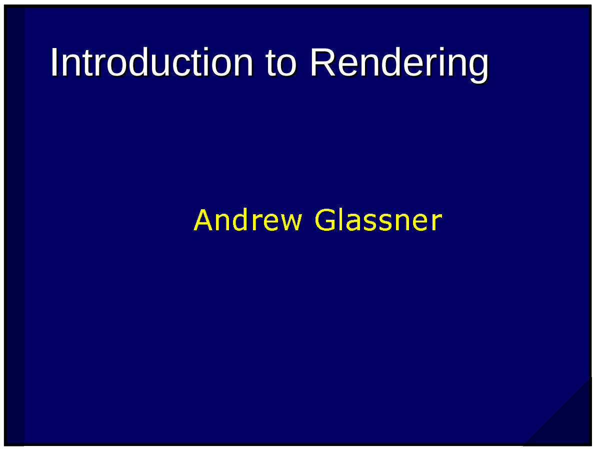
- Many primitives
- Many modelers
- Use what you need
  - modeler, primitive, construction style, and level of detail
- Think before you model!



An Introduction to Rendering

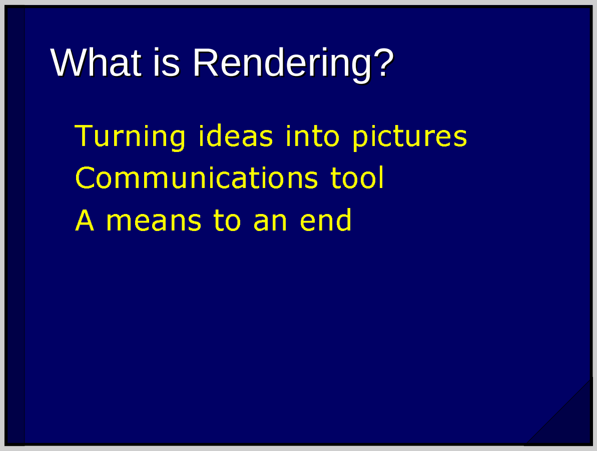
Andrew Glassner

SIGGRAPH 2001
EXPLORE INTERACTION AND DIGITAL IMAGES



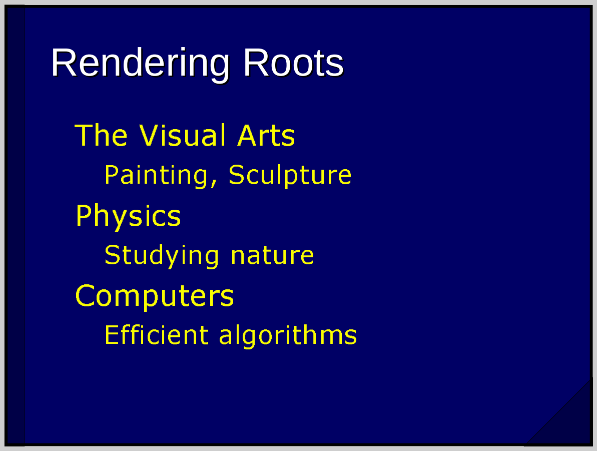
Introduction to Rendering

Andrew Glassner



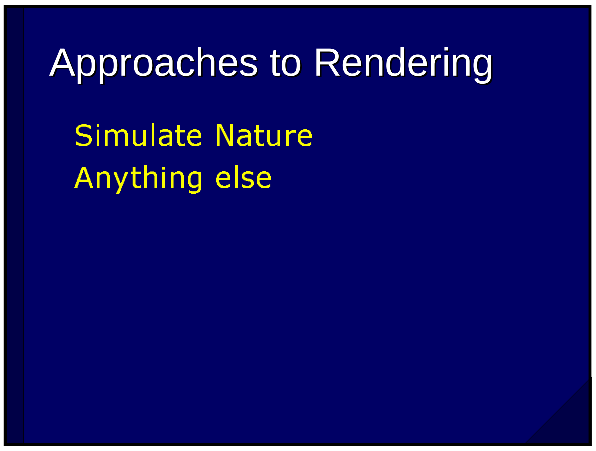
What is Rendering?

- Turning ideas into pictures
- Communications tool
- A means to an end



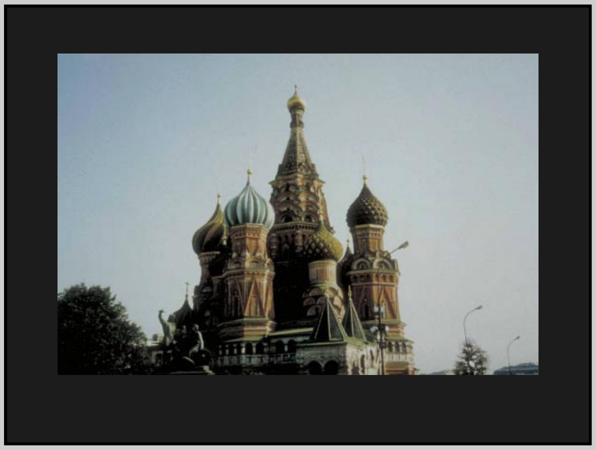
Rendering Roots

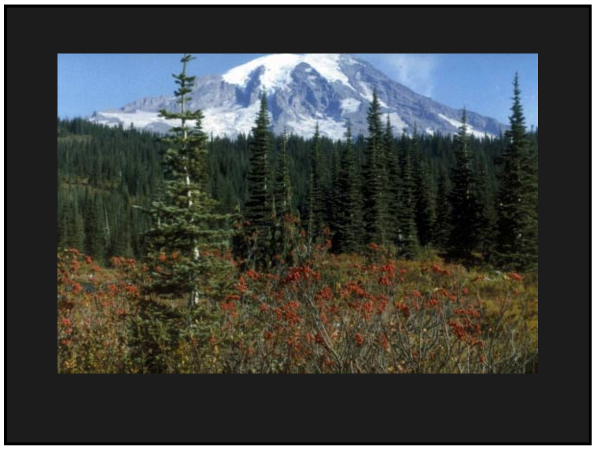
- The Visual Arts
  - Painting, Sculpture
- Physics
  - Studying nature
- Computers
  - Efficient algorithms

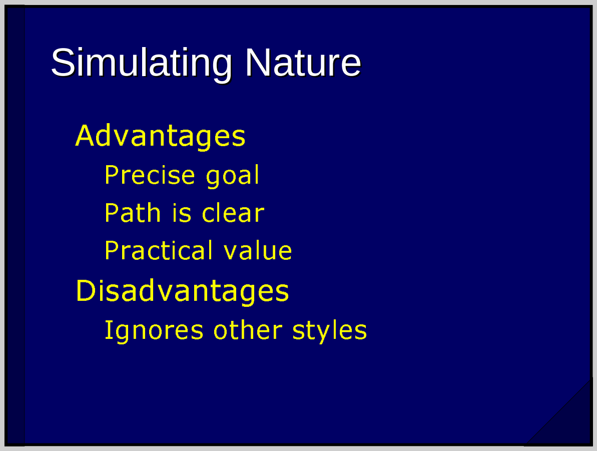


Approaches to Rendering

- Simulate Nature
- Anything else

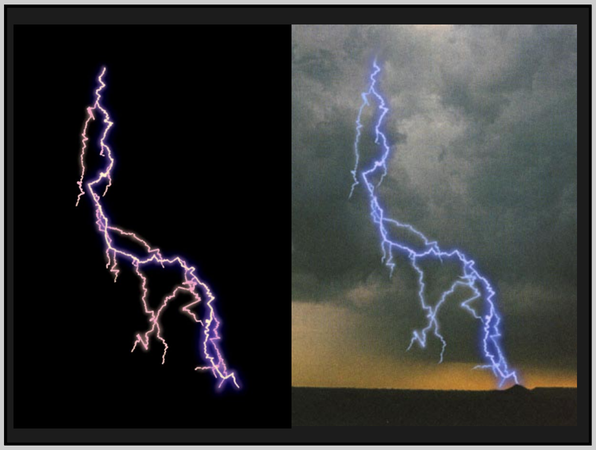


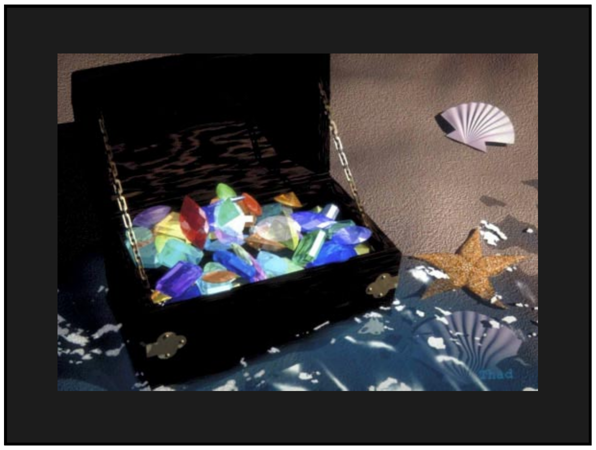


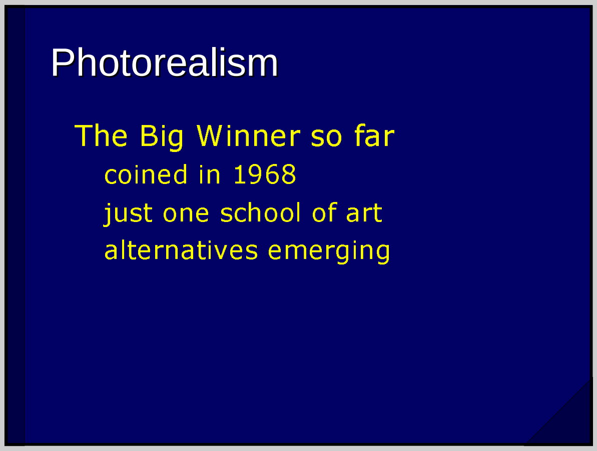


Simulating Nature

- Advantages
  - Precise goal
  - Path is clear
  - Practical value
- Disadvantages
  - Ignores other styles

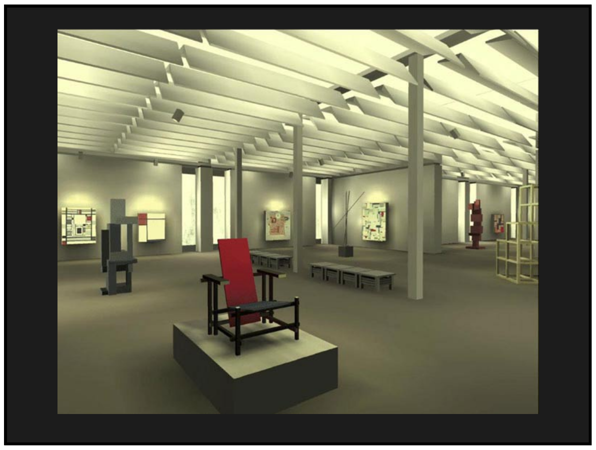


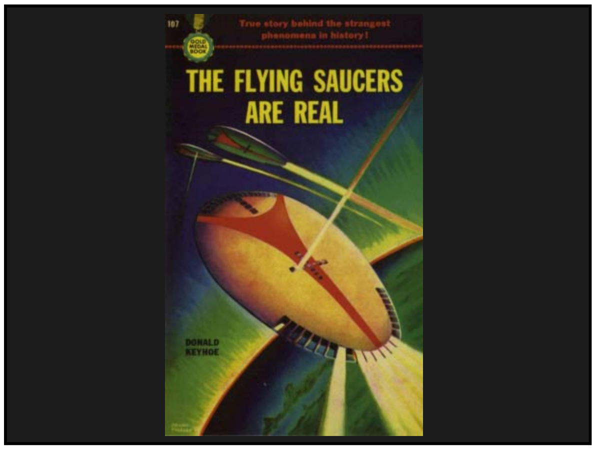


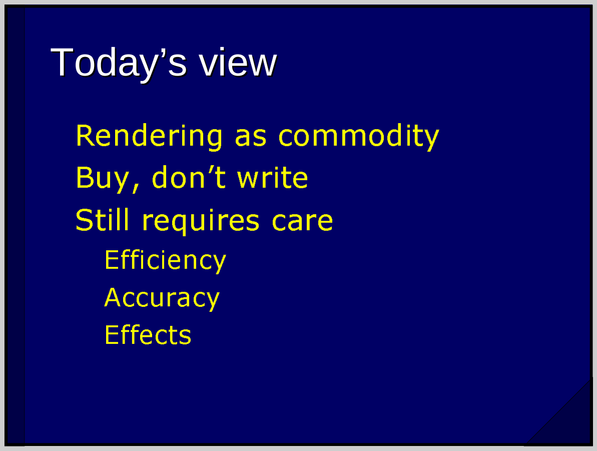


Photorealism

- The Big Winner so far
  - coined in 1968
  - just one school of art
  - alternatives emerging

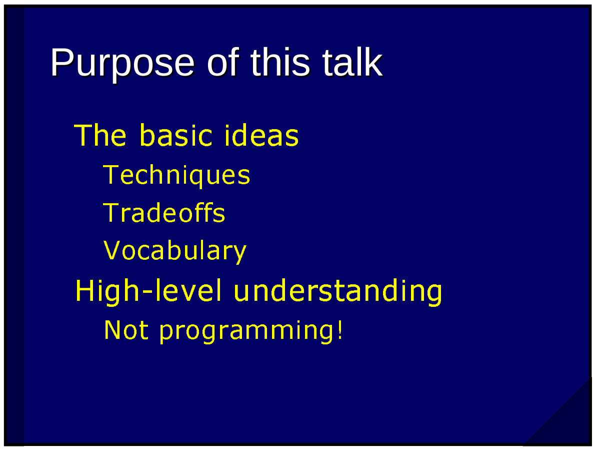



187
GOLD MEDAL BOOK
True story behind the strangest phenomena in history!
THE FLYING SAUCERS ARE REAL
DONALD KEYHOE



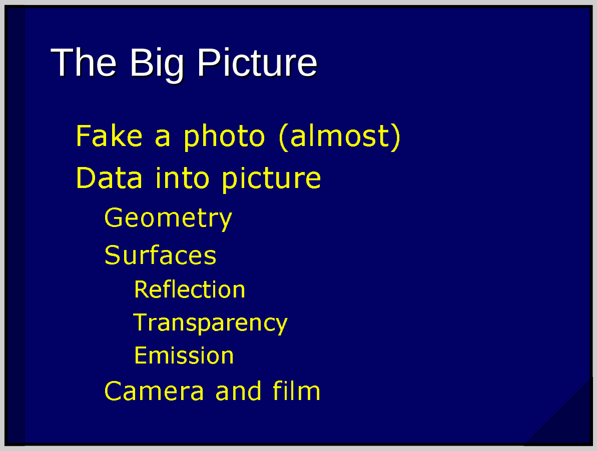
Today's view

- Rendering as commodity
- Buy, don't write
- Still requires care
  - Efficiency
  - Accuracy
  - Effects



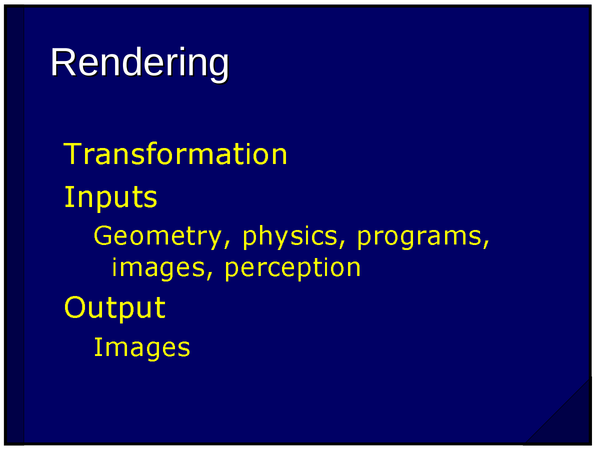
Purpose of this talk

- The basic ideas
  - Techniques
  - Tradeoffs
  - Vocabulary
- High-level understanding
  - Not programming!



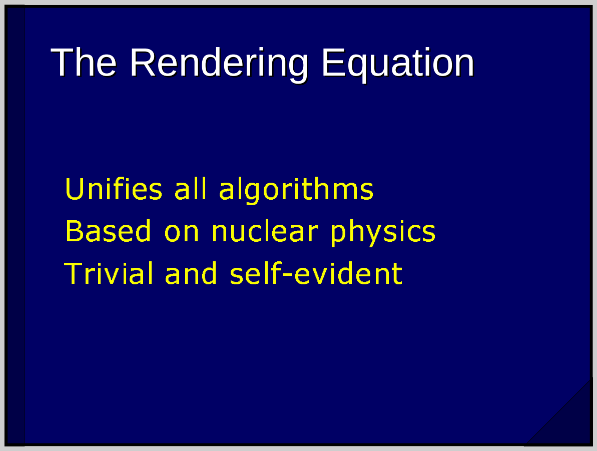
The Big Picture

- Fake a photo (almost)
- Data into picture
  - Geometry
  - Surfaces
    - Reflection
    - Transparency
    - Emission
  - Camera and film



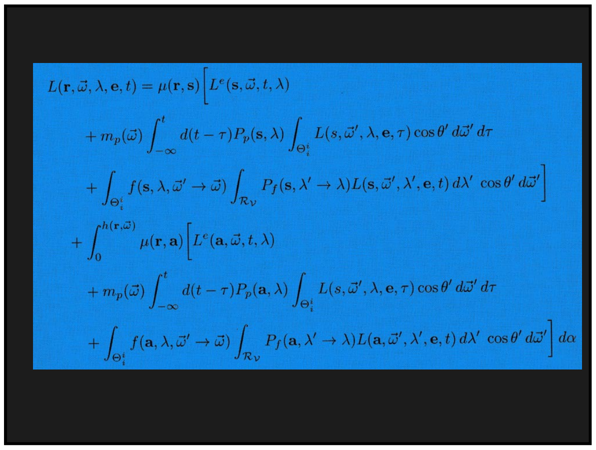
Rendering

- Transformation
- Inputs
  - Geometry, physics, programs, images, perception
- Output
  - Images

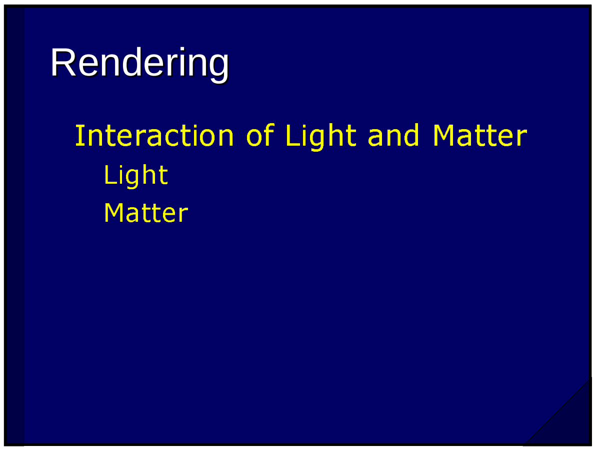


The Rendering Equation

- Unifies all algorithms
- Based on nuclear physics
- Trivial and self-evident

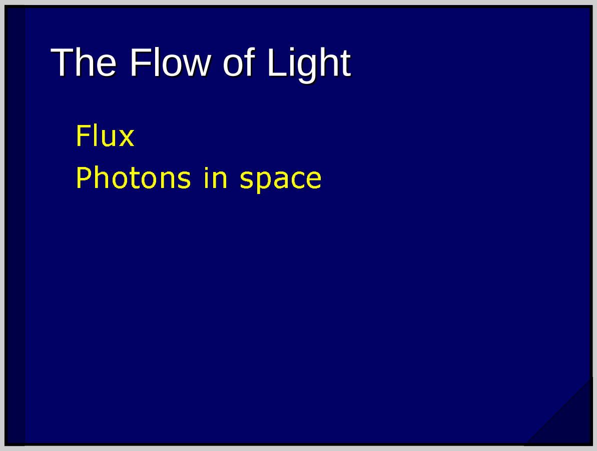


$$
\begin{aligned}
L(\mathbf{r},\vec{\omega},\lambda,\mathbf{e},t) = \mu(\mathbf{r},\mathbf{s})\Big[& L^e(\mathbf{s},\vec{\omega},t,\lambda) \\
& + m_p(\vec{\omega}) \int_{-\infty}^{t} d(t-\tau) P_p(\mathbf{s},\lambda) \int_{\Theta_i^i} L(s,\vec{\omega}',\lambda,\mathbf{e},\tau)\cos\theta'\, d\vec{\omega}'\, d\tau \\
& + \int_{\Theta_i^i} f(\mathbf{s},\lambda,\vec{\omega}' \to \vec{\omega}) \int_{\mathcal{R}_\mathcal{V}} P_f(\mathbf{s},\lambda' \to \lambda) L(\mathbf{s},\vec{\omega}',\lambda',\mathbf{e},t)\, d\lambda' \ \cos\theta'\, d\vec{\omega}' \Big] \\
+ \int_0^{h(\mathbf{r},\vec{\omega})} \mu(\mathbf{r},\mathbf{a}) \Big[& L^e(\mathbf{a},\vec{\omega},t,\lambda) \\
& + m_p(\vec{\omega}) \int_{-\infty}^{t} d(t-\tau) P_p(\mathbf{a},\lambda) \int_{\Theta_i^i} L(s,\vec{\omega}',\lambda,\mathbf{e},\tau)\cos\theta'\, d\vec{\omega}'\, d\tau \\
& + \int_{\Theta_i^i} f(\mathbf{a},\lambda,\vec{\omega}' \to \vec{\omega}) \int_{\mathcal{R}_\mathcal{V}} P_f(\mathbf{a},\lambda' \to \lambda) L(\mathbf{a},\vec{\omega}',\lambda',\mathbf{e},t)\, d\lambda' \ \cos\theta'\, d\vec{\omega}' \Big]\, d\alpha
\end{aligned}
$$



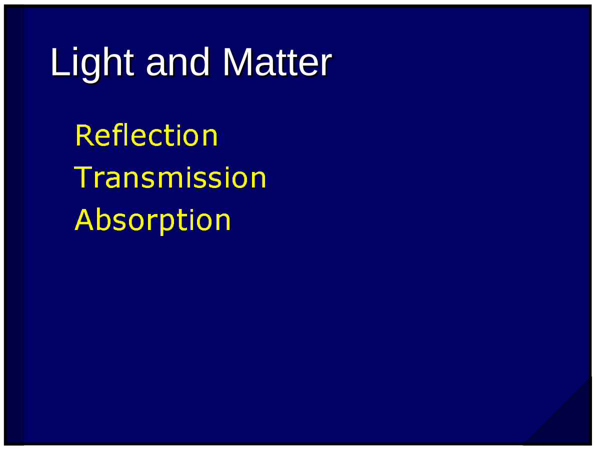
Rendering

- Interaction of Light and Matter
  - Light
  - Matter



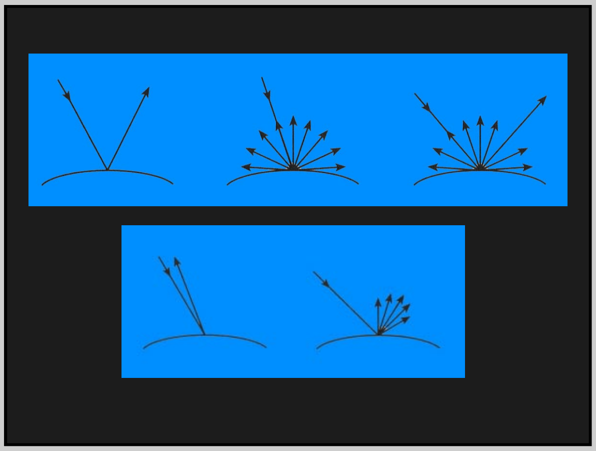
The Flow of Light

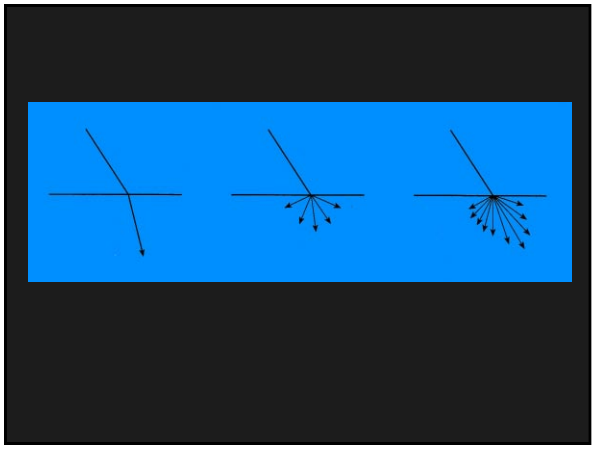
- Flux
- Photons in space

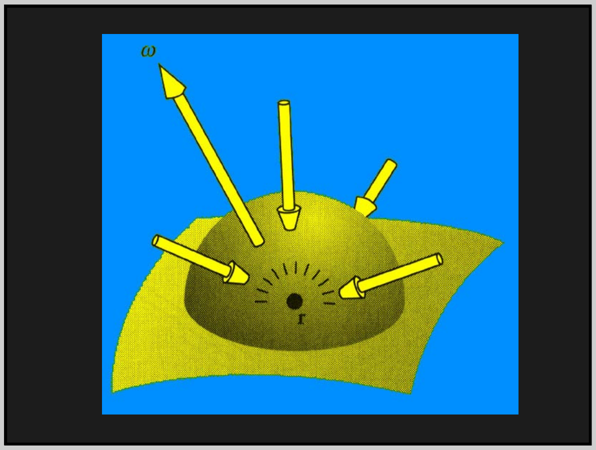


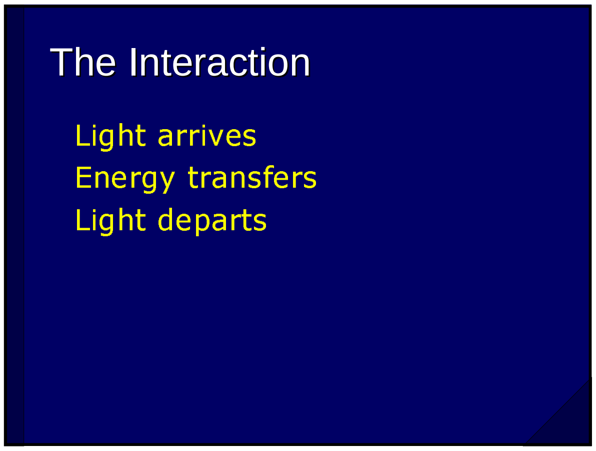
Light and Matter

- Reflection
- Transmission
- Absorption



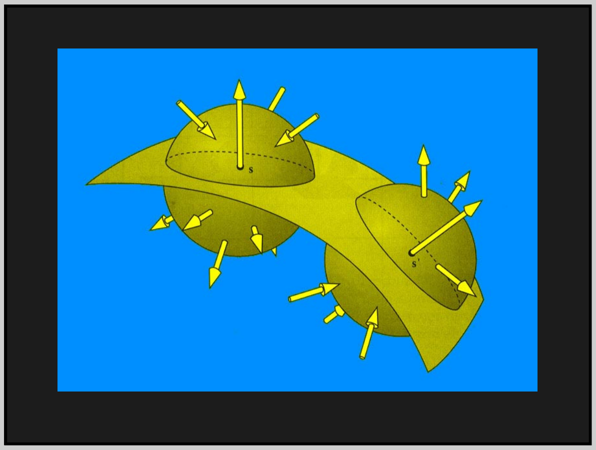


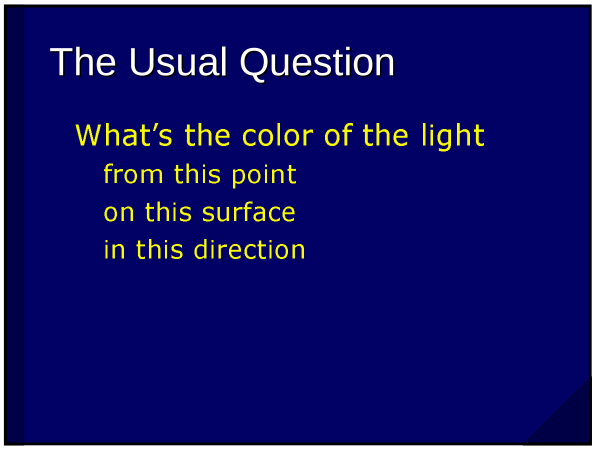




The Interaction

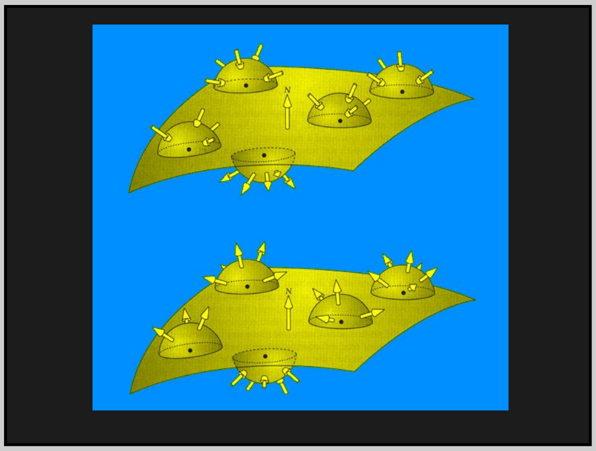
- Light arrives
- Energy transfers
- Light departs

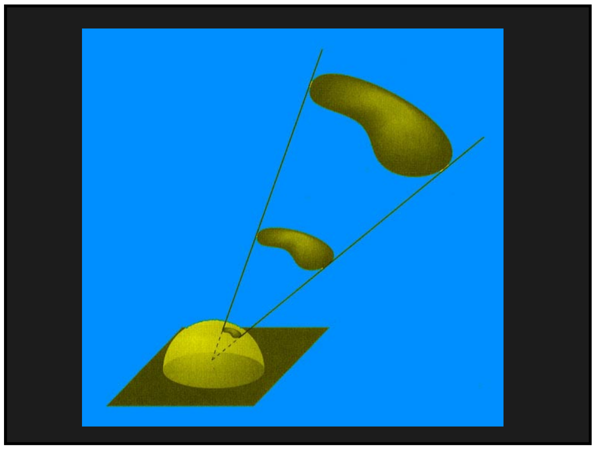


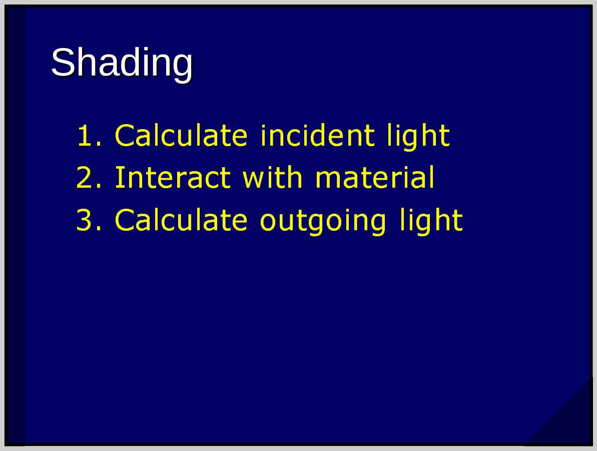


The Usual Question

What's the color of the light
- from this point
- on this surface
- in this direction

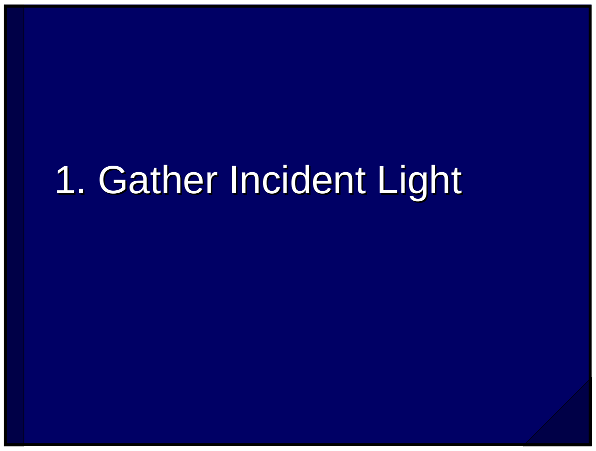




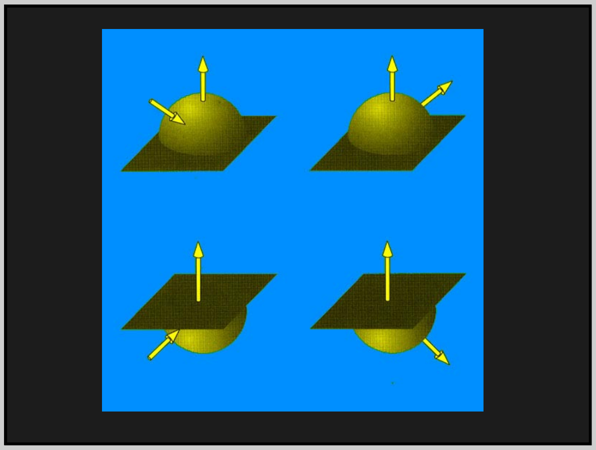


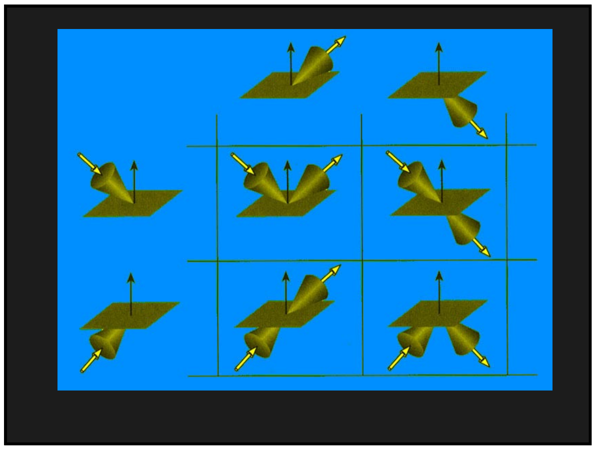
Shading

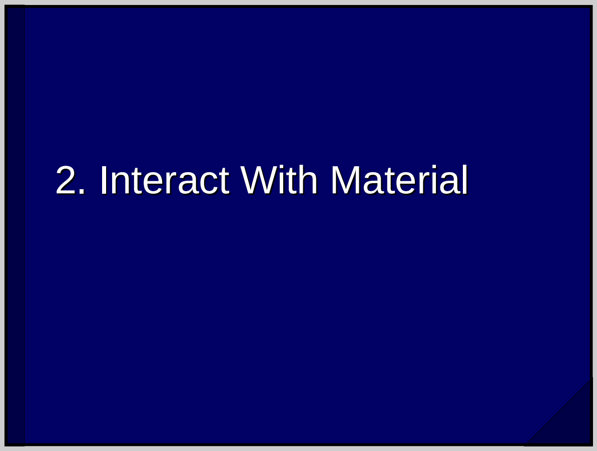
1. Calculate incident light
2. Interact with material
3. Calculate outgoing light



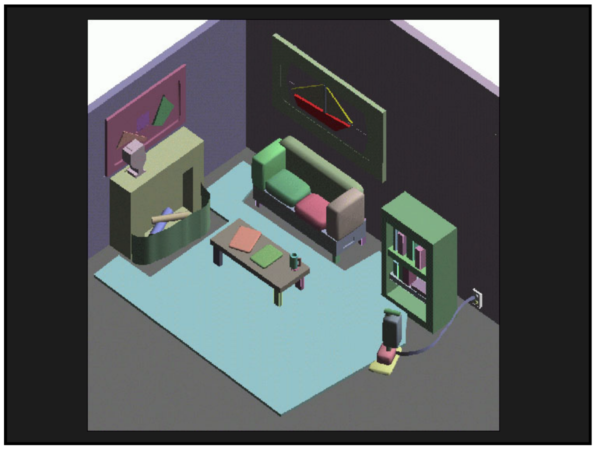
1. Gather Incident Light

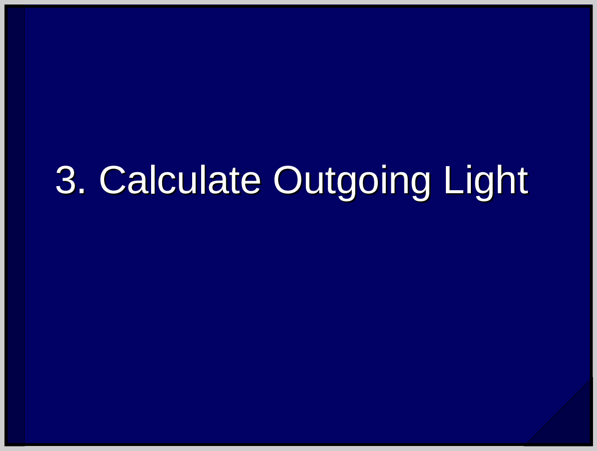




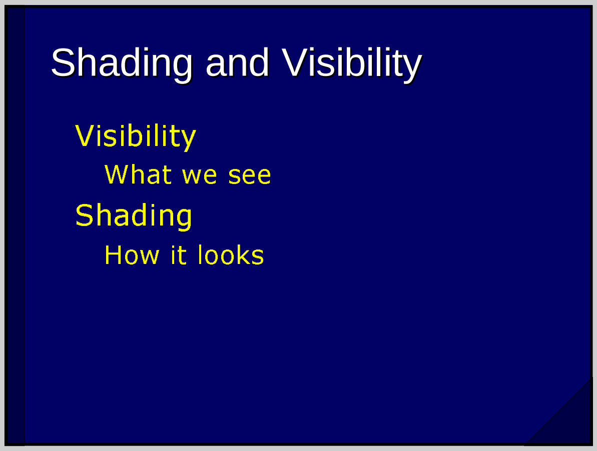


2. Interact With Material



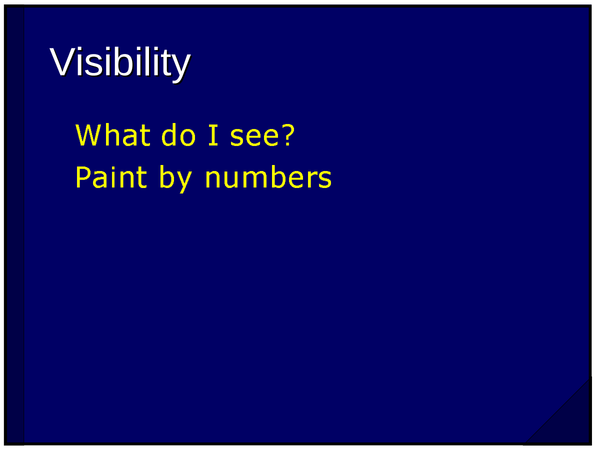


3. Calculate Outgoing Light



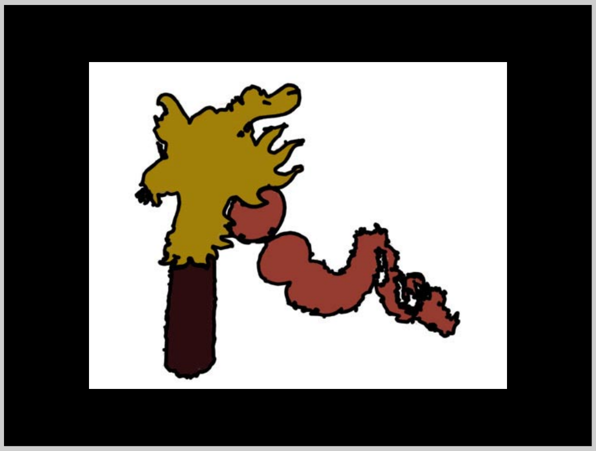
Shading and Visibility

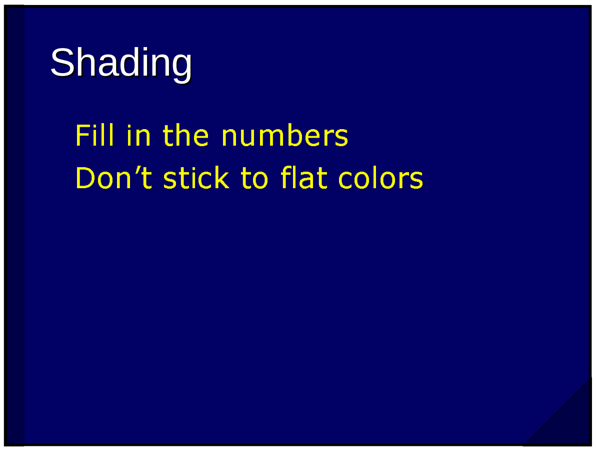
- Visibility
  - What we see
- Shading
  - How it looks



Visibility

- What do I see?
- Paint by numbers

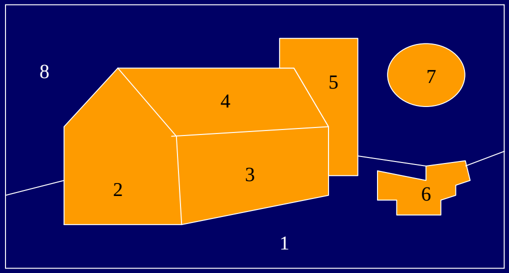




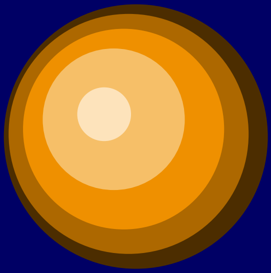
Shading

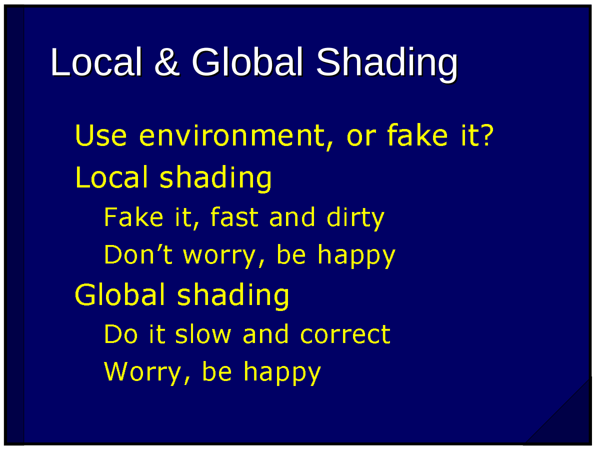
- Fill in the numbers
- Don't stick to flat colors

Visibility: Paint-by-numbers



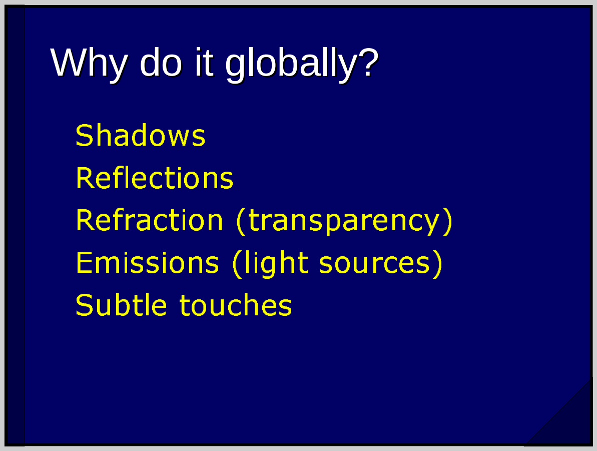
Shading: Object colors





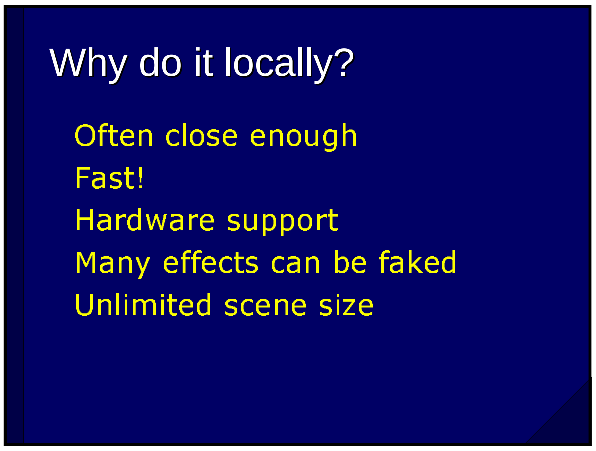
Local & Global Shading

- Use environment, or fake it?
- Local shading
  - Fake it, fast and dirty
  - Don’t worry, be happy
- Global shading
  - Do it slow and correct
  - Worry, be happy



Why do it globally?

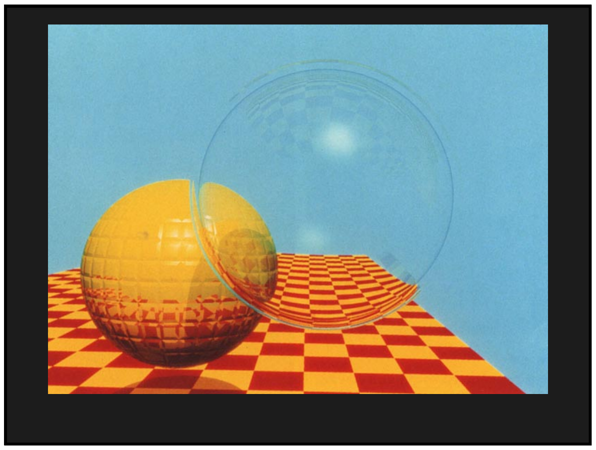
- Shadows
- Reflections
- Refraction (transparency)
- Emissions (light sources)
- Subtle touches

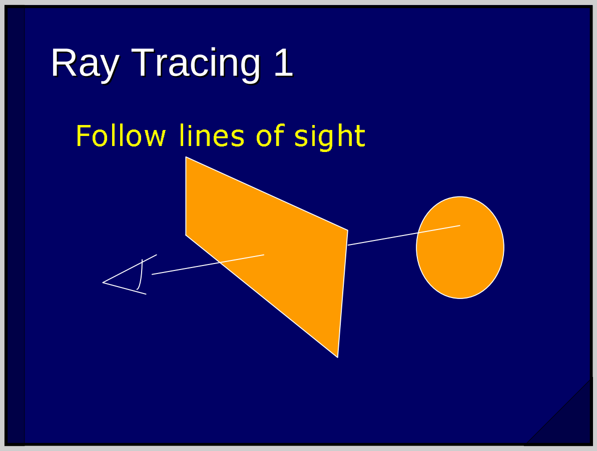


Why do it locally?

- Often close enough
- Fast!
- Hardware support
- Many effects can be faked
- Unlimited scene size

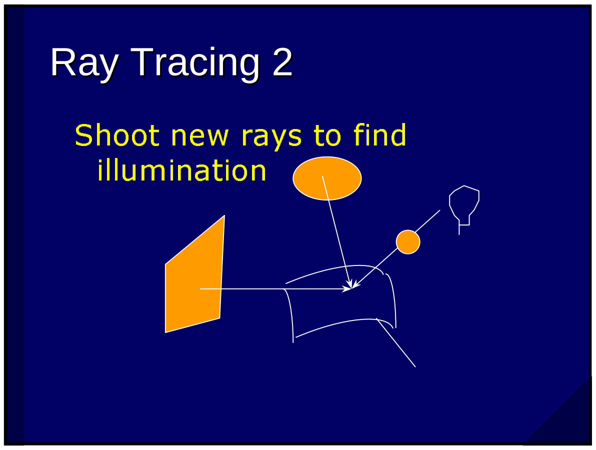
Ray Tracing





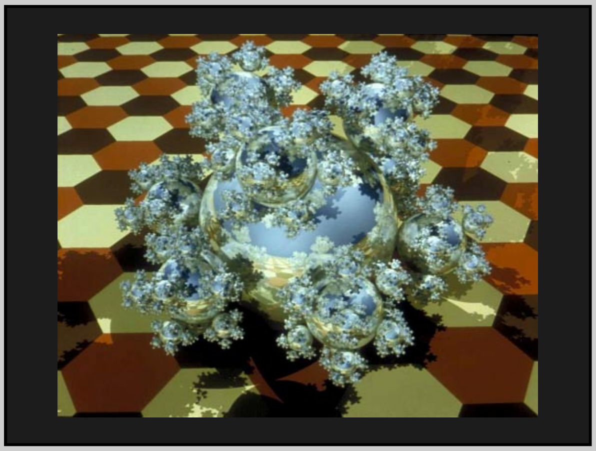
Ray Tracing 1

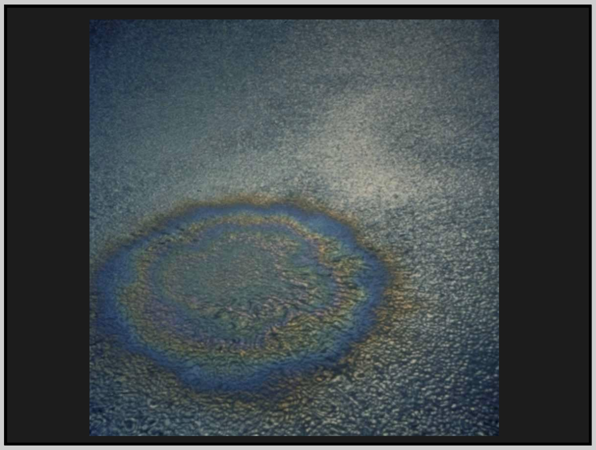
Follow lines of sight

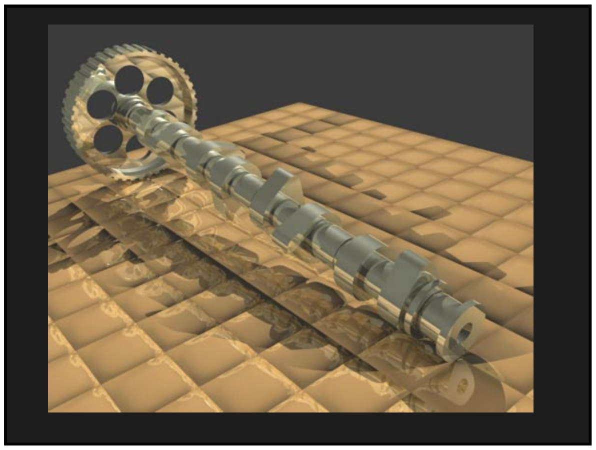


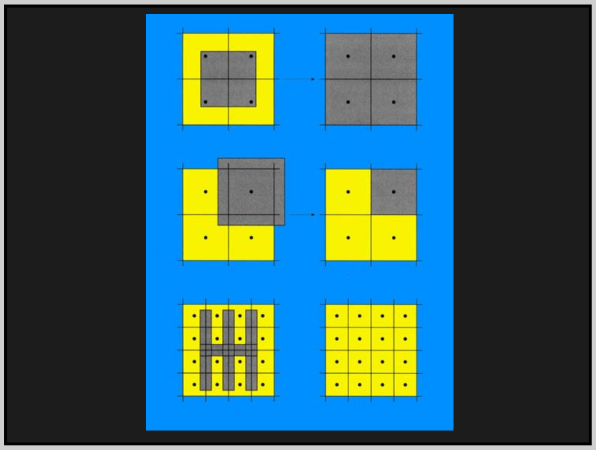
Ray Tracing 2

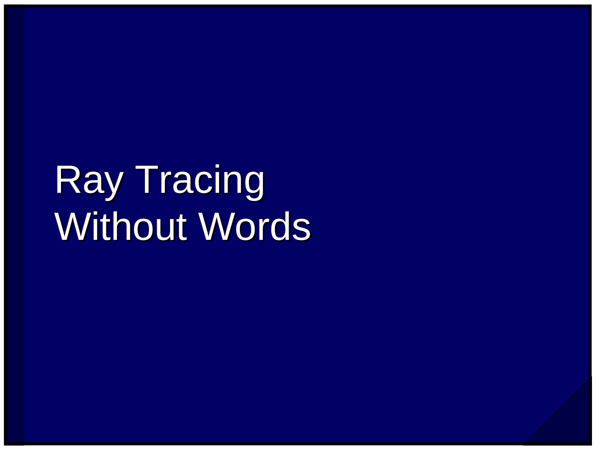
Shoot new rays to find illumination



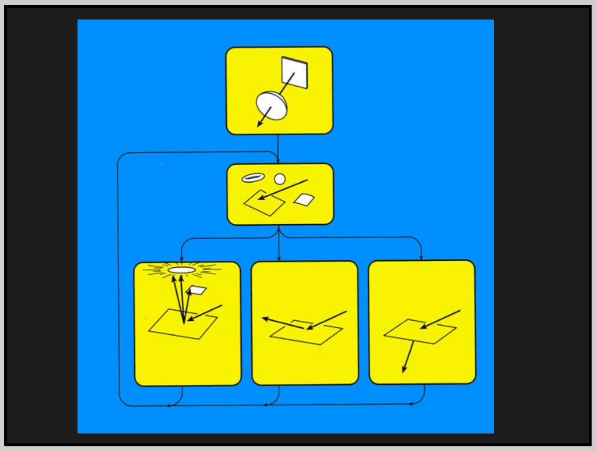


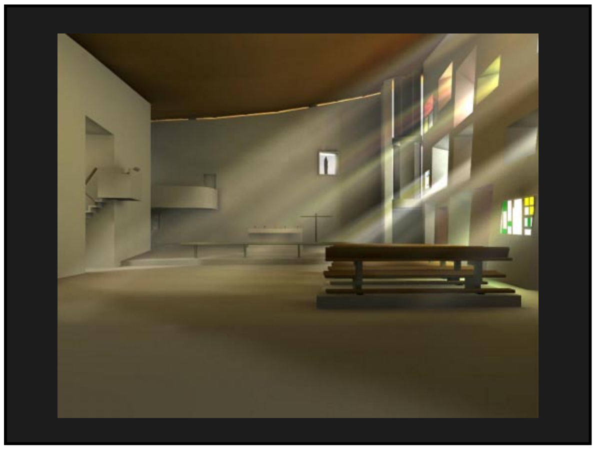


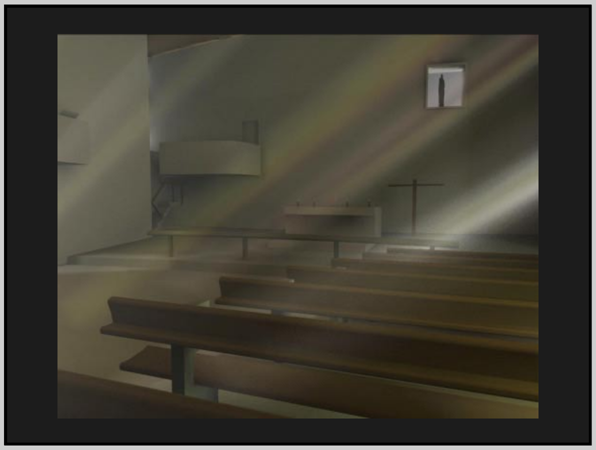


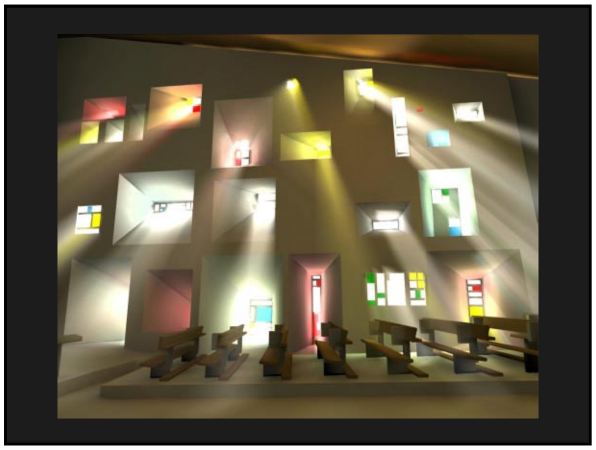


Ray Tracing
Without Words

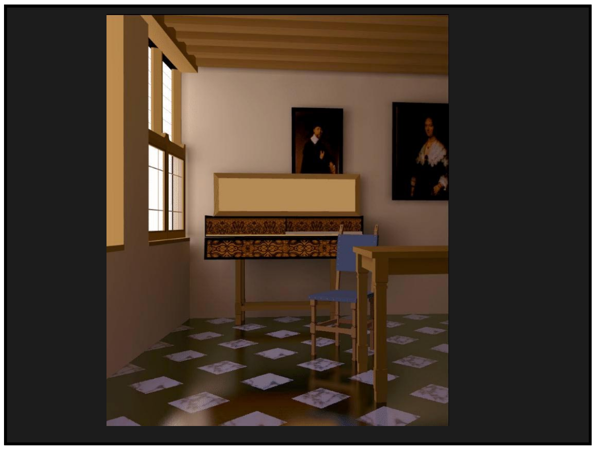


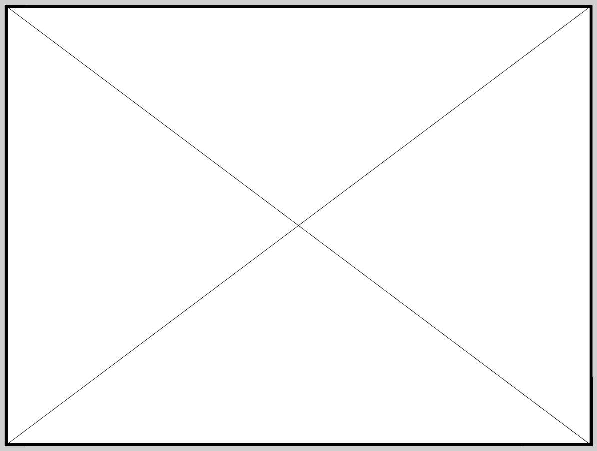


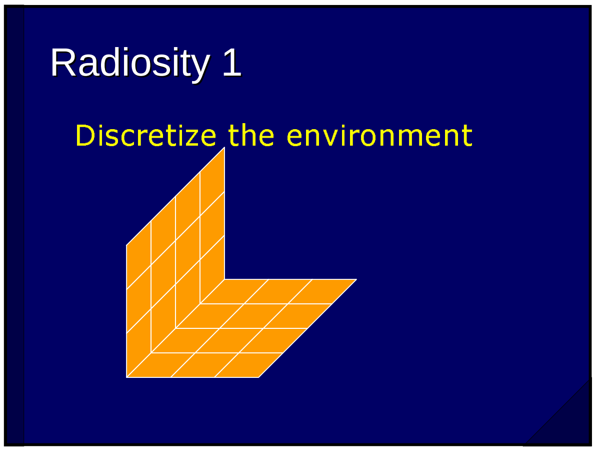




Radiosity

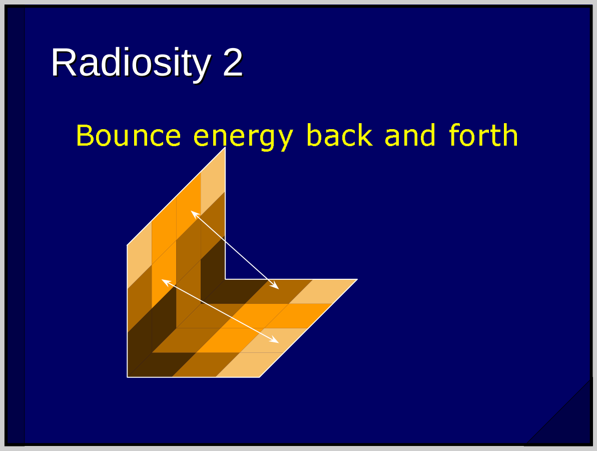






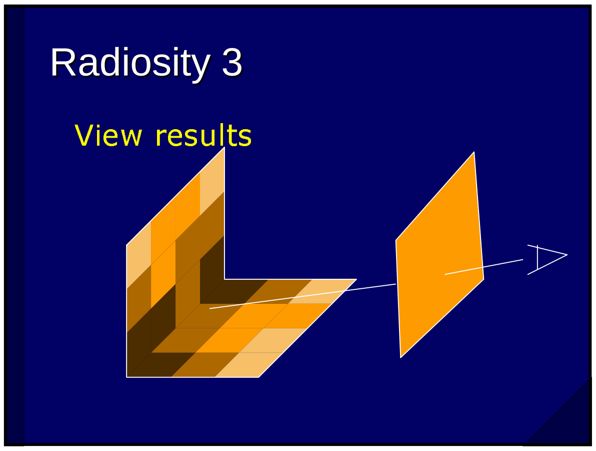
Radiosity 1

Discretize the environment



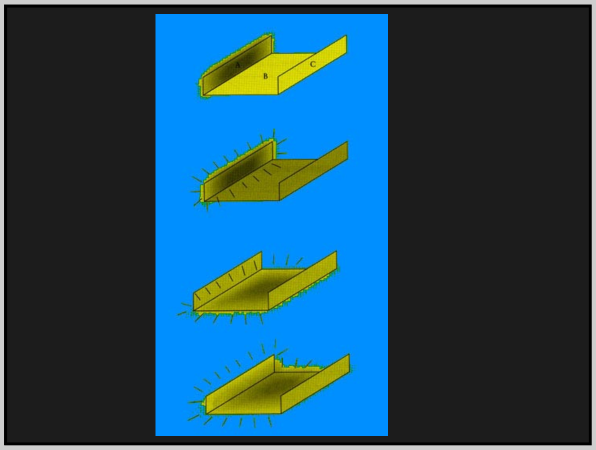
Radiosity 2

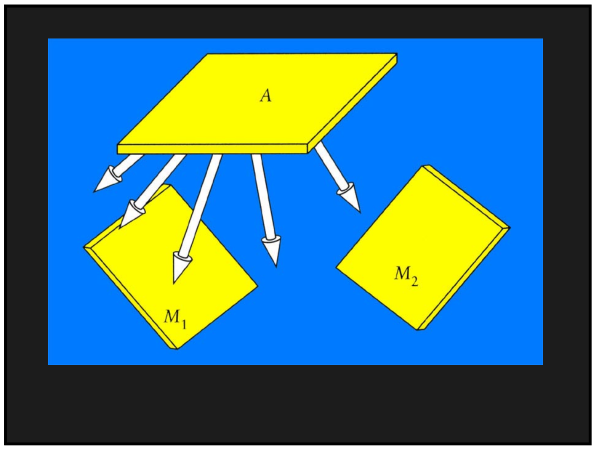
Bounce energy back and forth

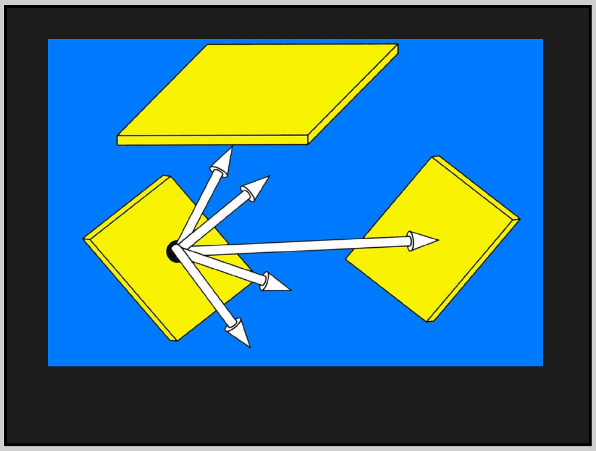


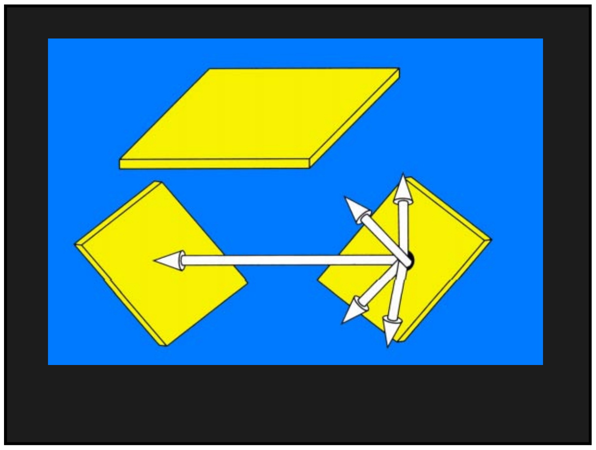
Radiosity 3

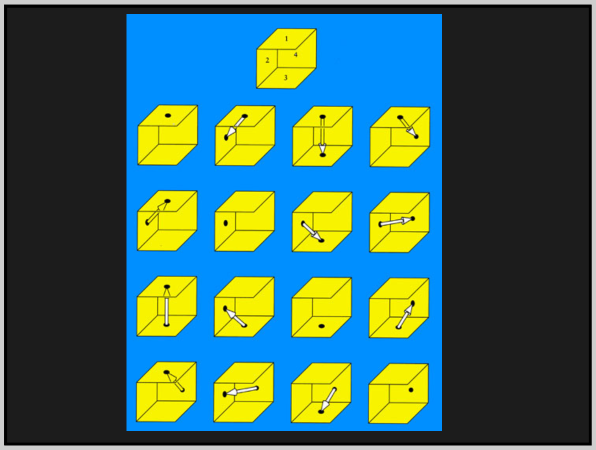
View results

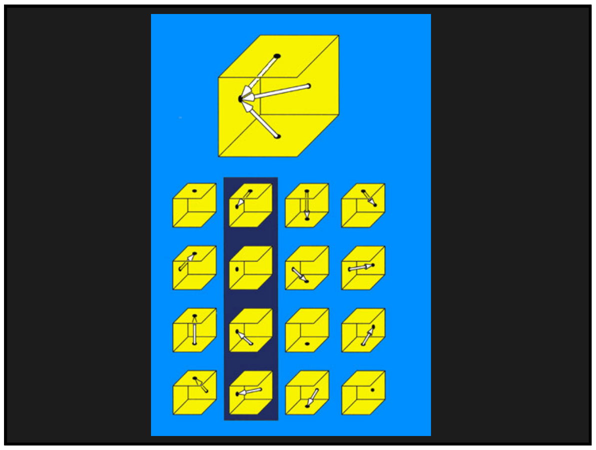


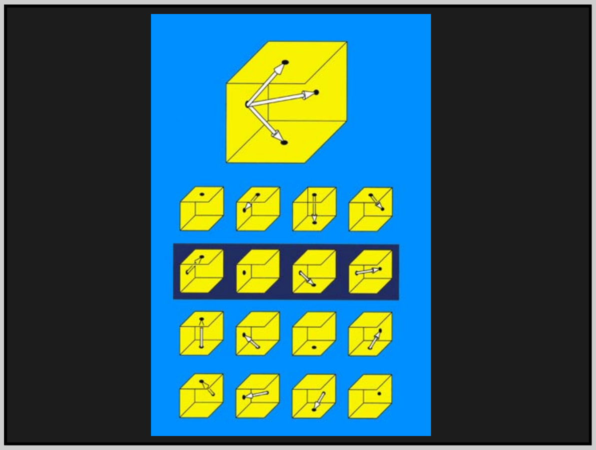


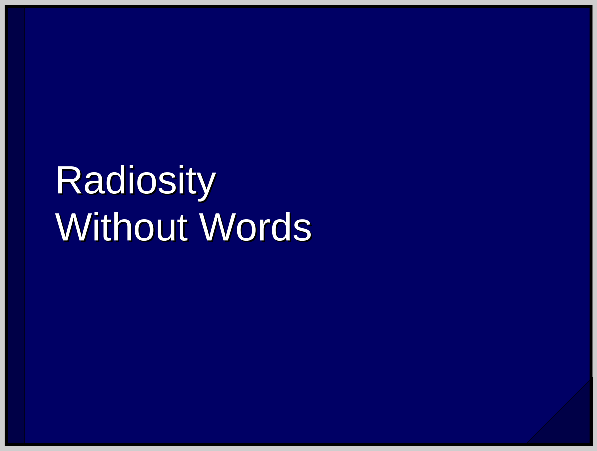




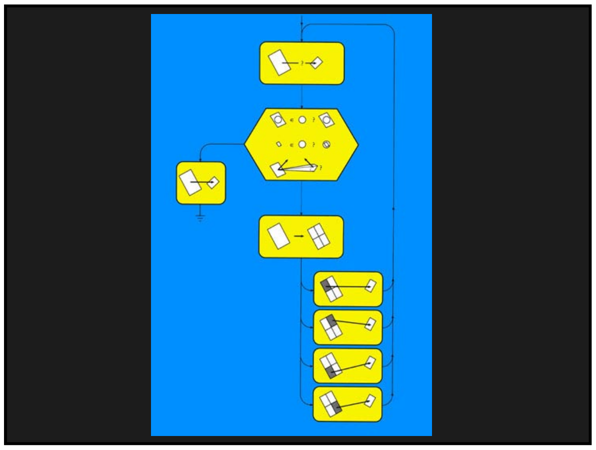


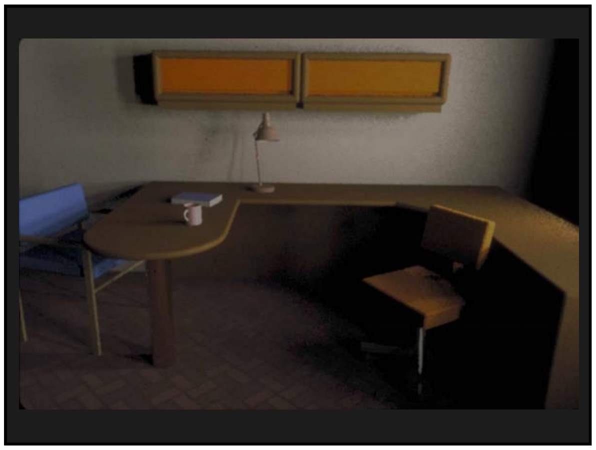


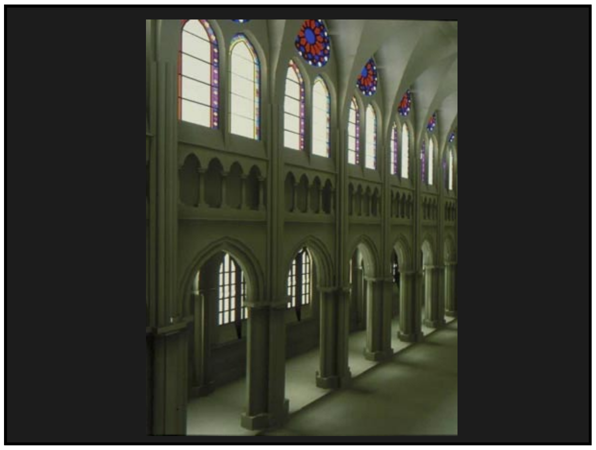


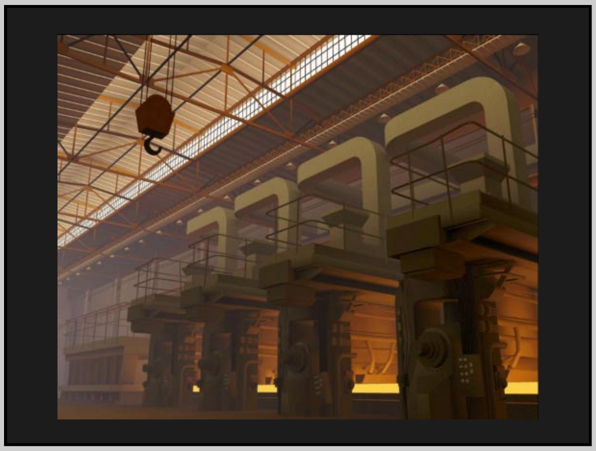


Radiosity
Without Words

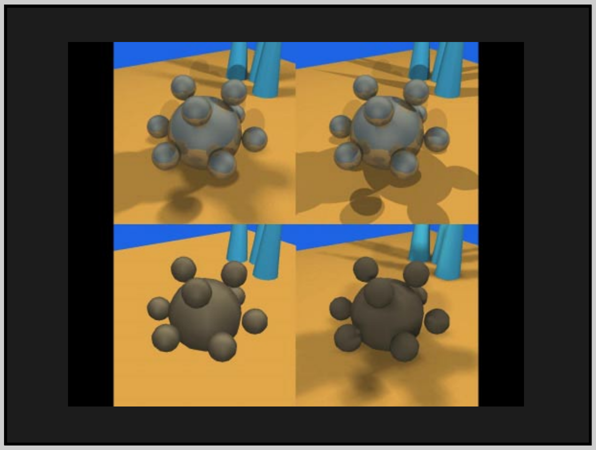


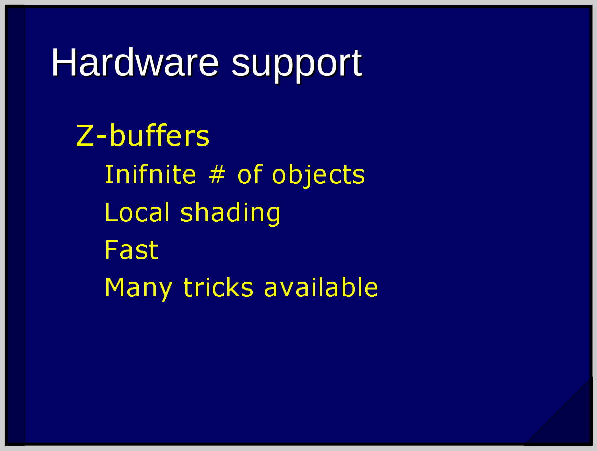






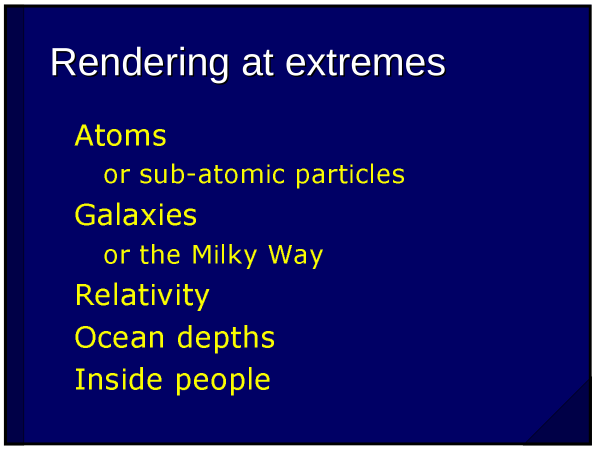
Comparing
Ray Tracing and
Radiosity





Hardware support

- Z-buffers
  - Inifnite # of objects
  - Local shading
  - Fast
  - Many tricks available



Rendering at extremes

- Atoms
  - or sub-atomic particles
- Galaxies
  - or the Milky Way
- Relativity
- Ocean depths
- Inside people

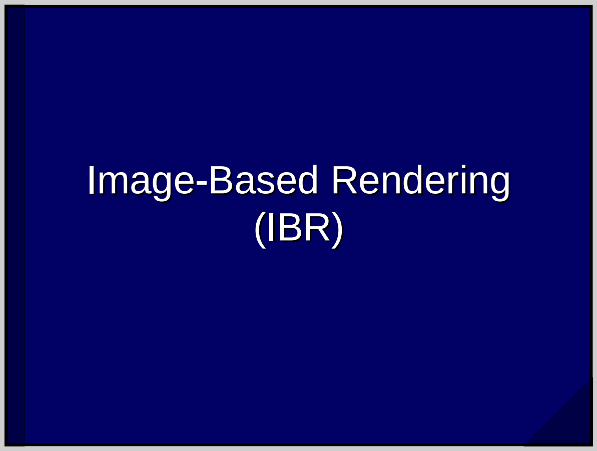


Image-Based Rendering (IBR)

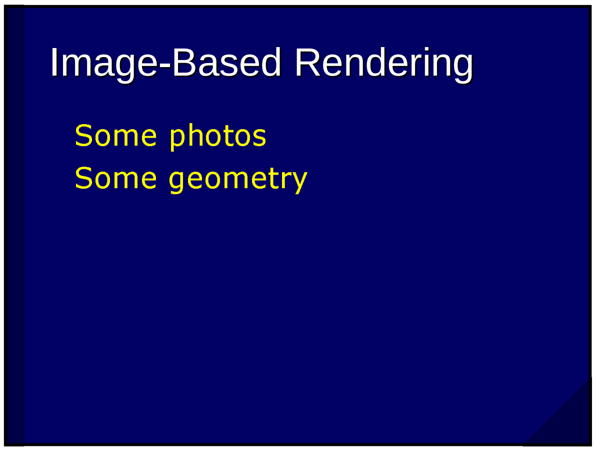
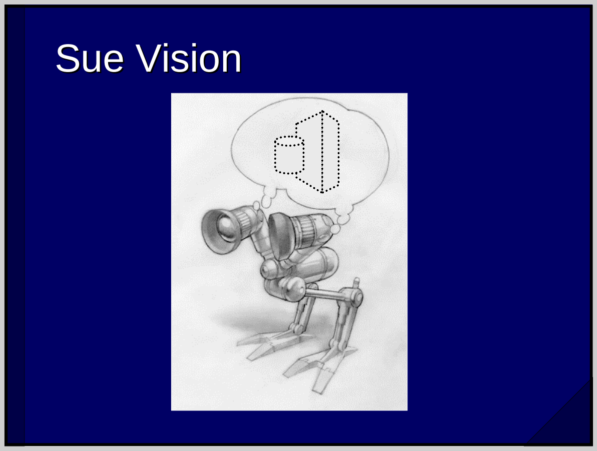
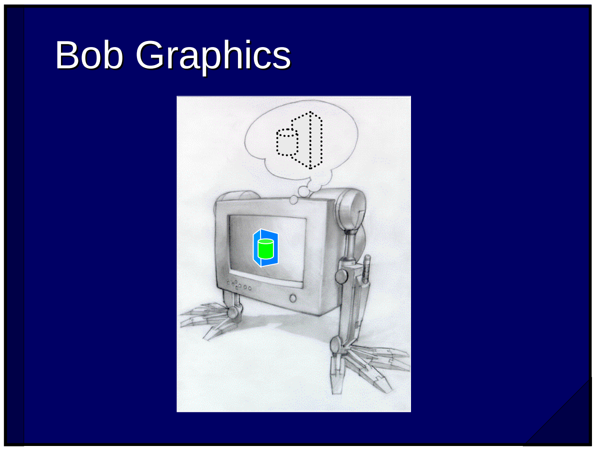


Image-Based Rendering

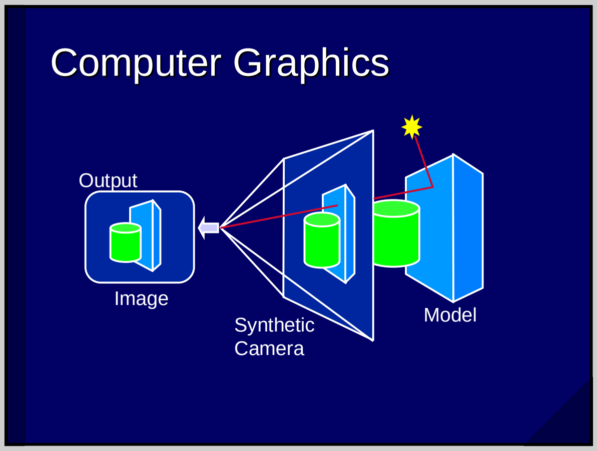
- Some photos
- Some geometry



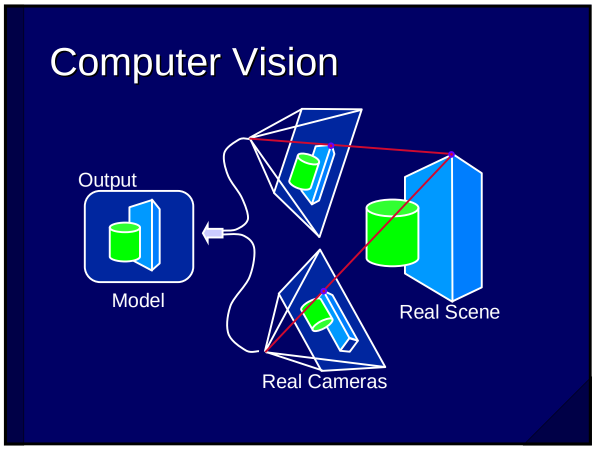
Sue Vision



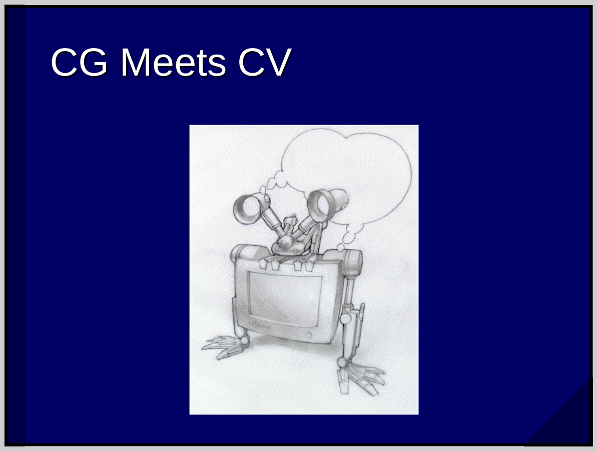
Bob Graphics



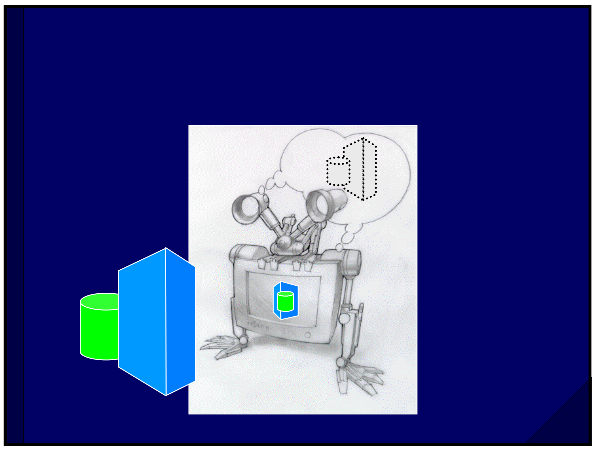
Computer Graphics

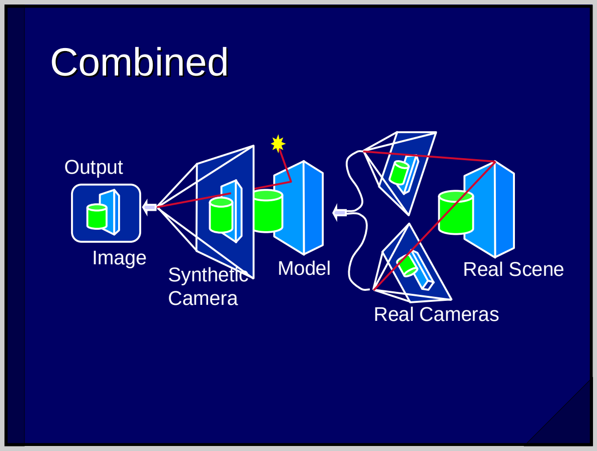


Computer Vision

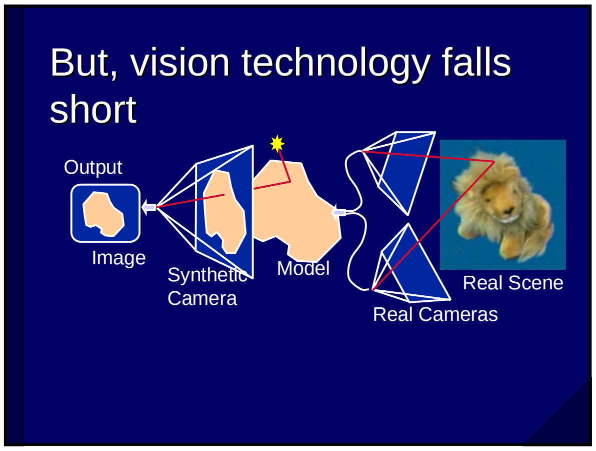


CG Meets CV

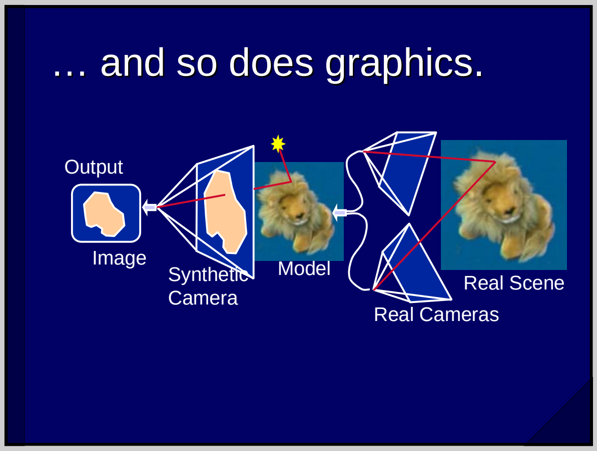




Combined



But, vision technology falls short



… and so does graphics.

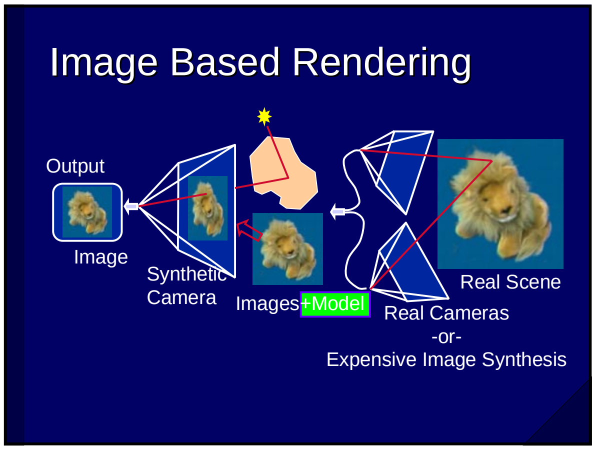
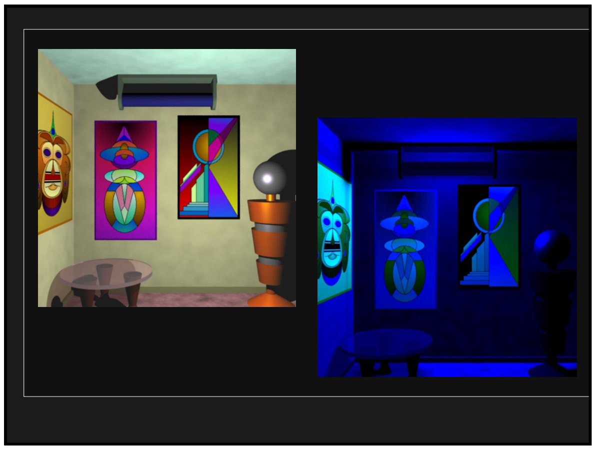
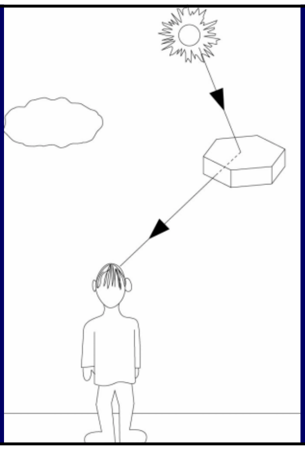


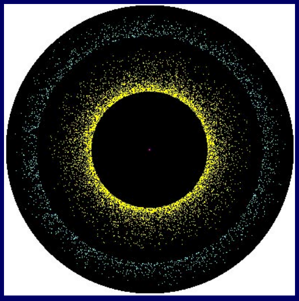
Image Based Rendering

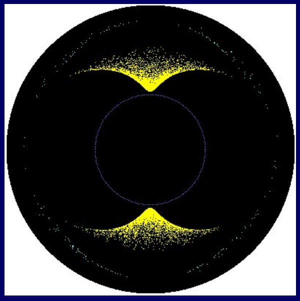
Fluorescence

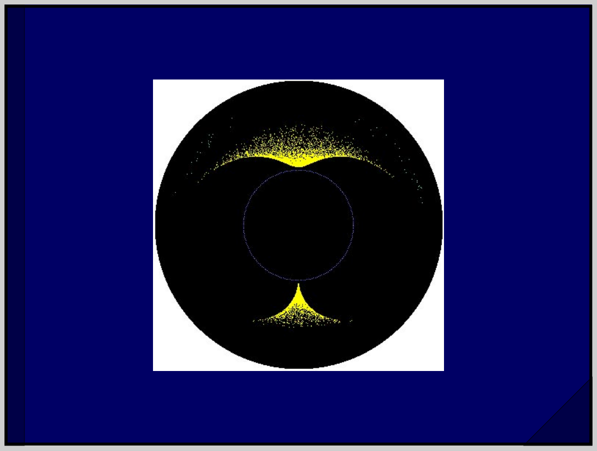


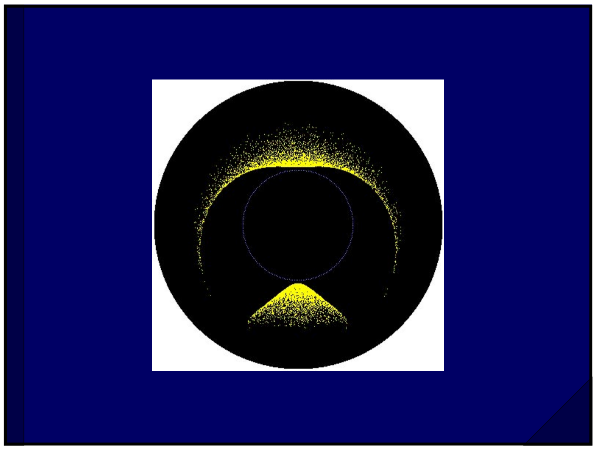
Solar Halos

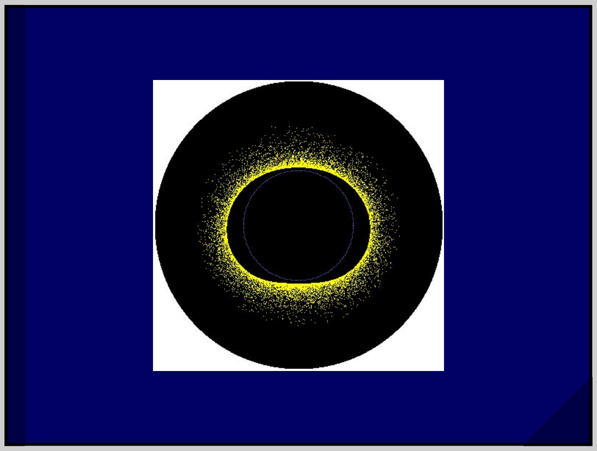


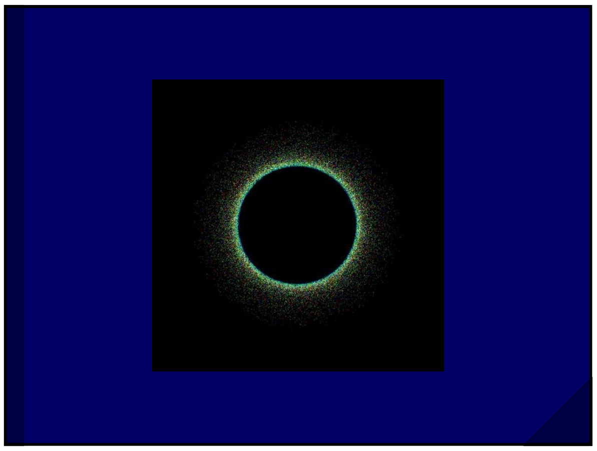


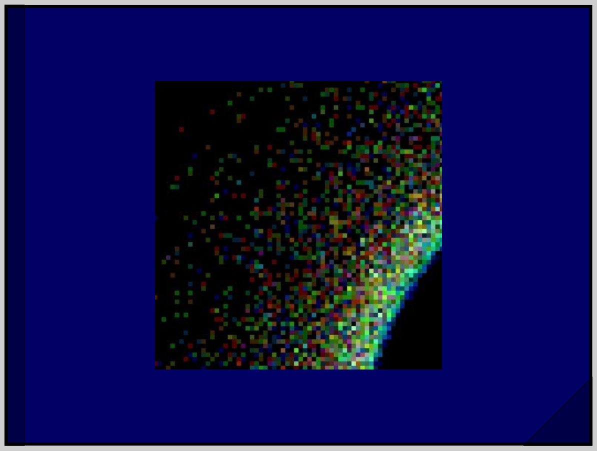


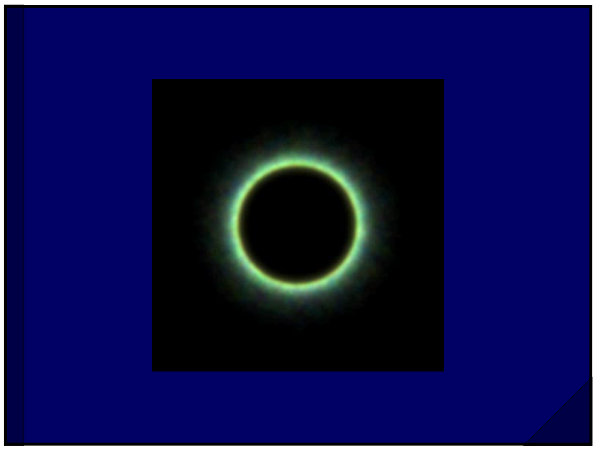


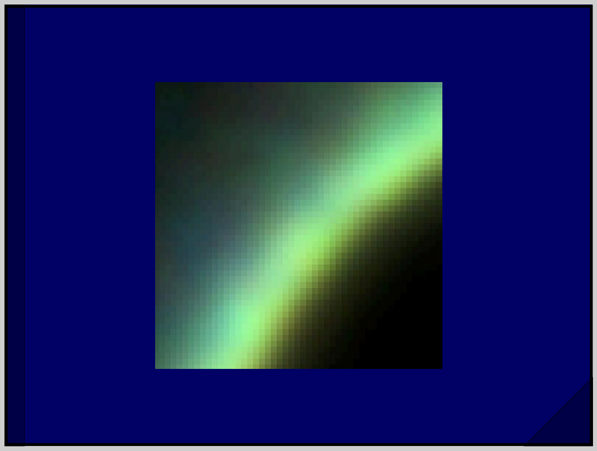


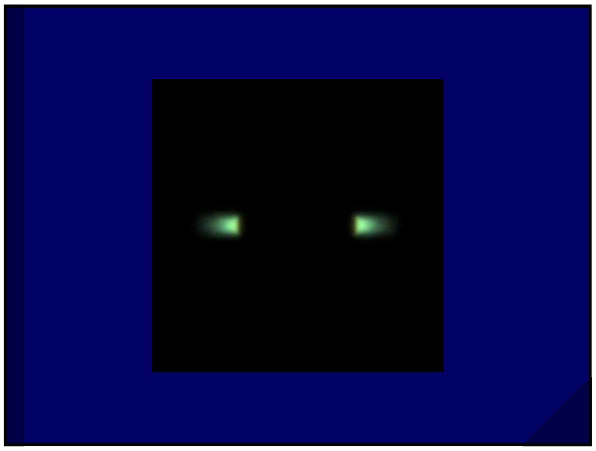


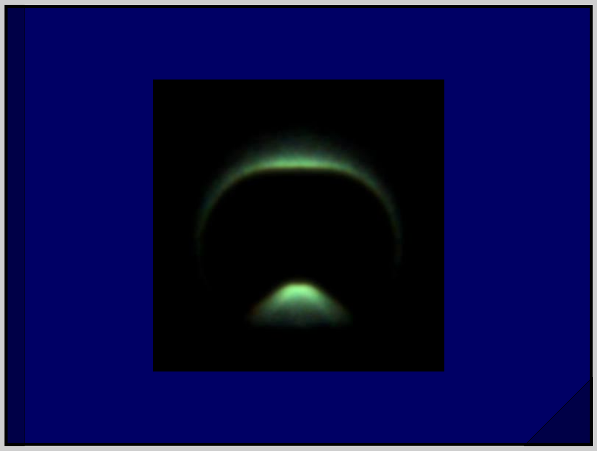


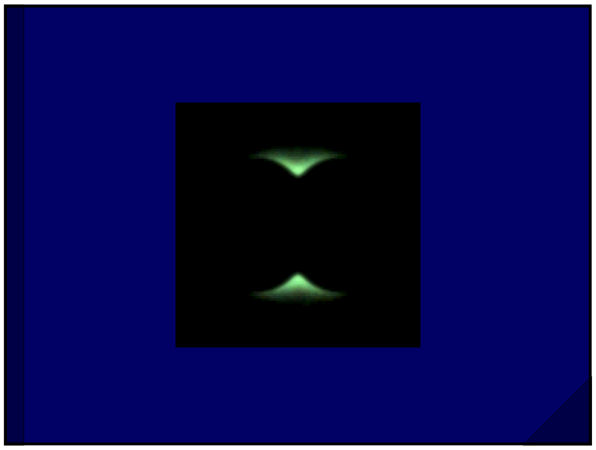


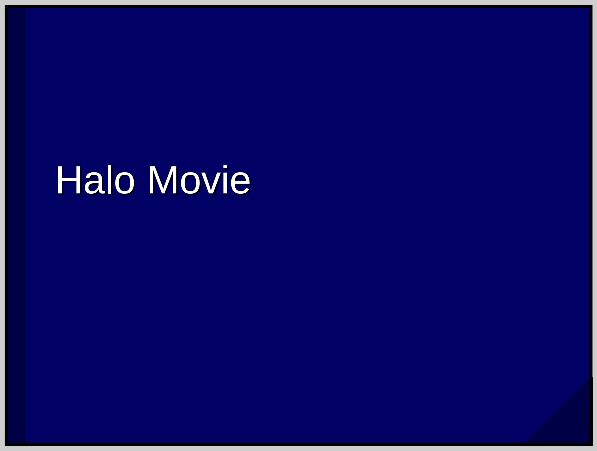




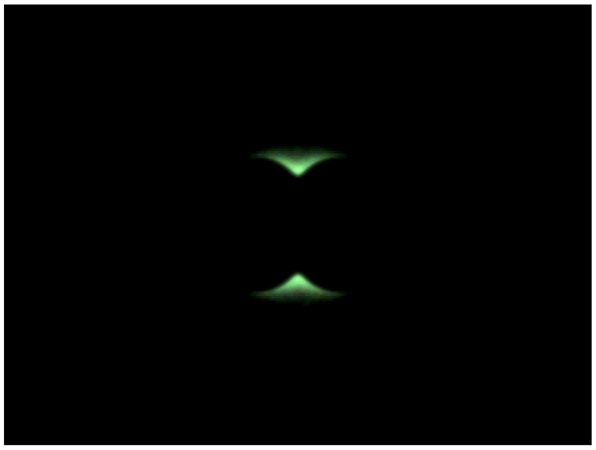




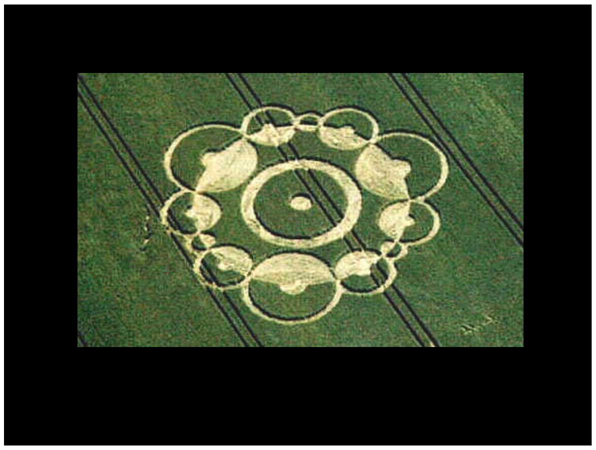


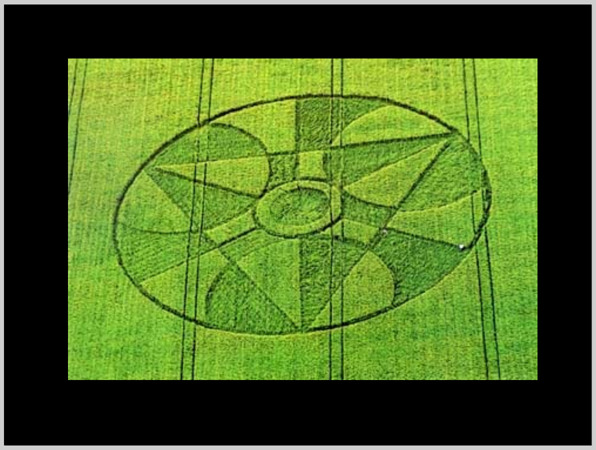


Halo Movie

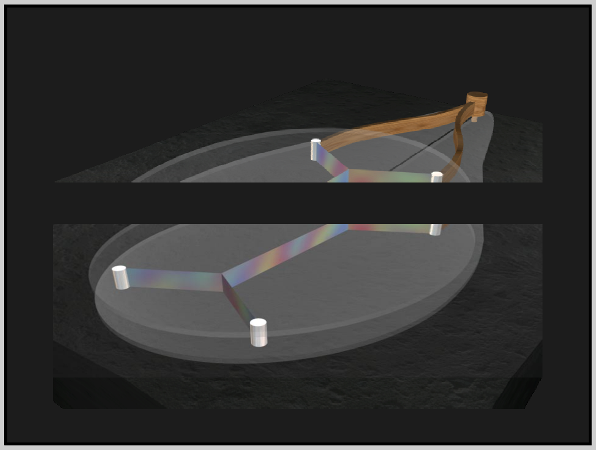


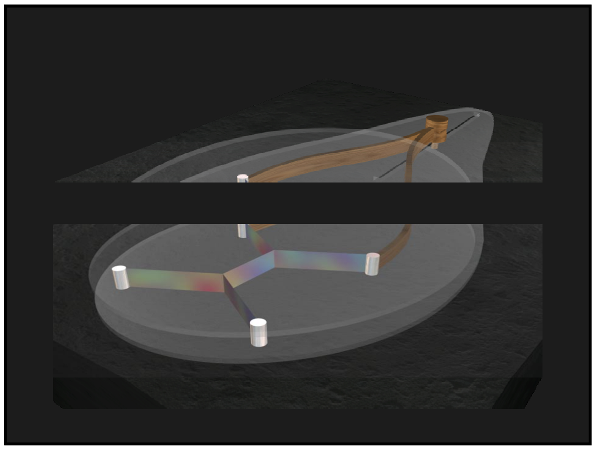
Odd Output Media

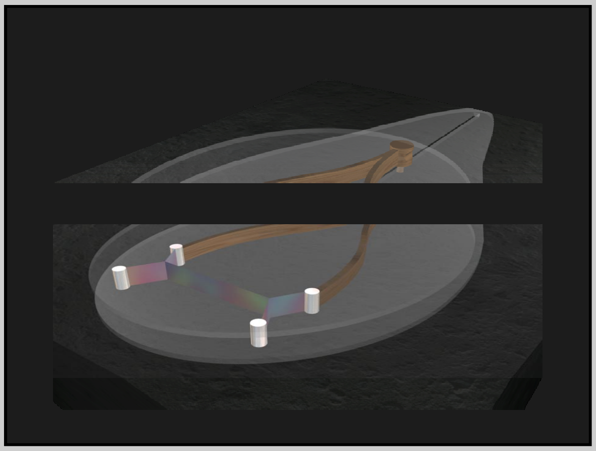


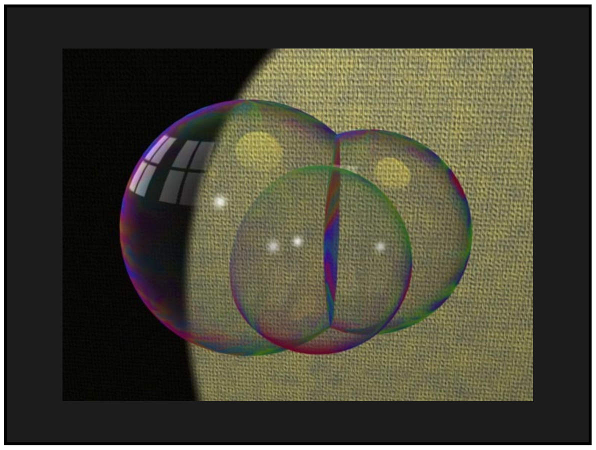


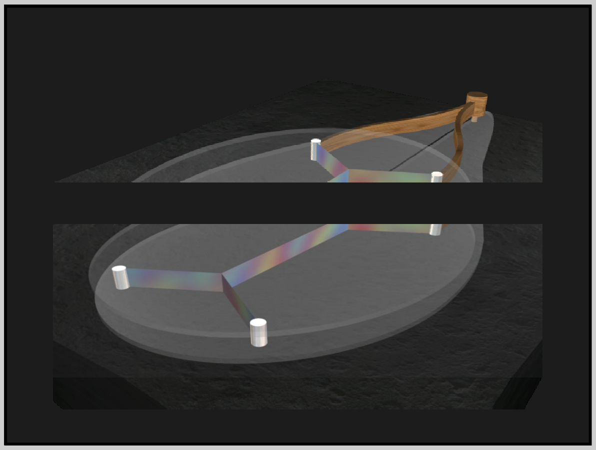
Interference

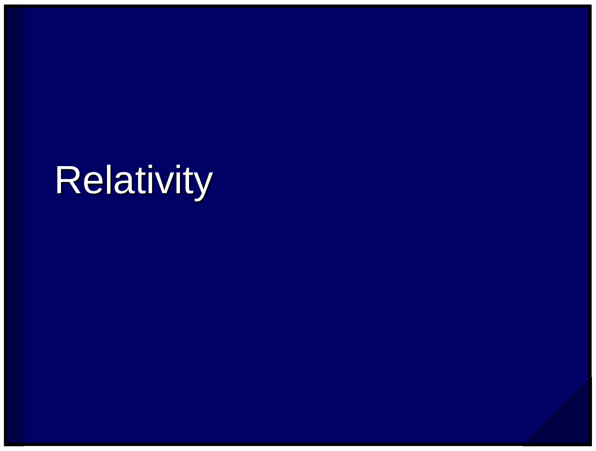




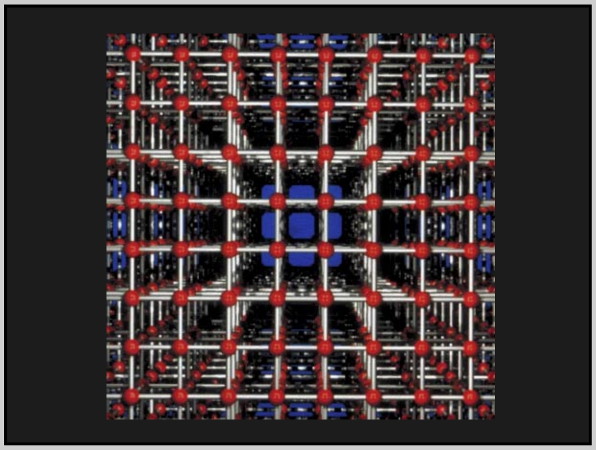


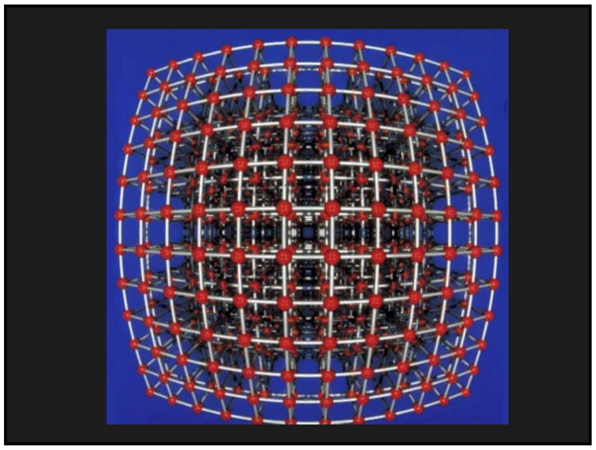


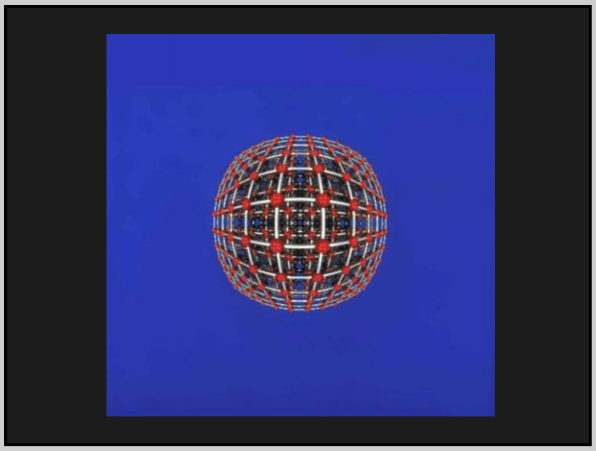




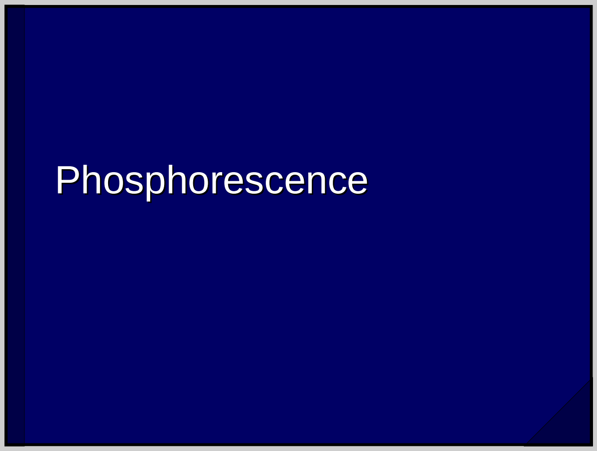
Relativity



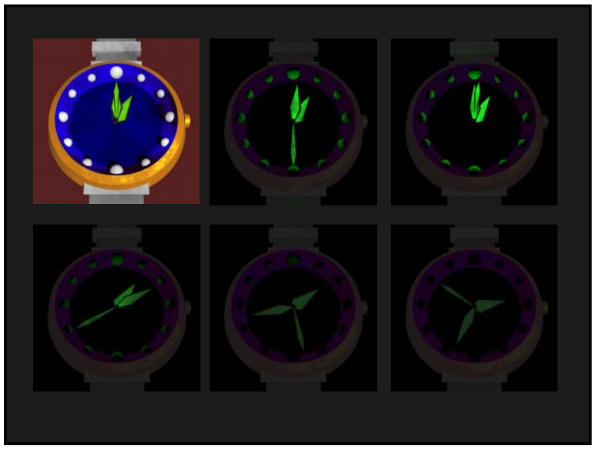


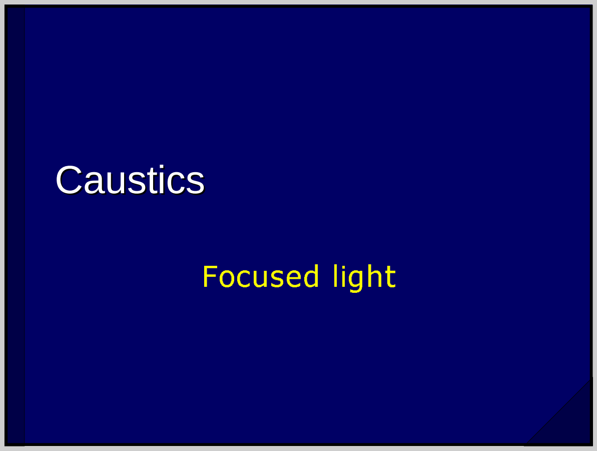


Polarization



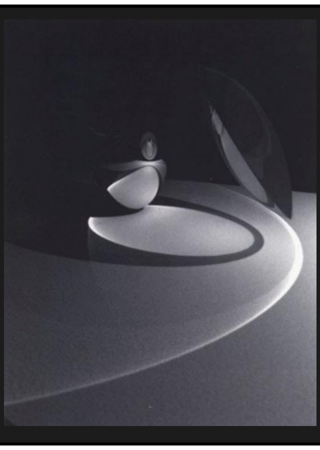
Phosphorescence

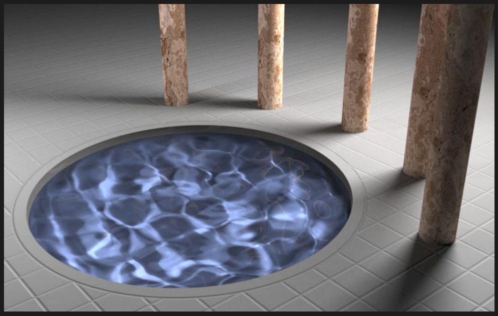


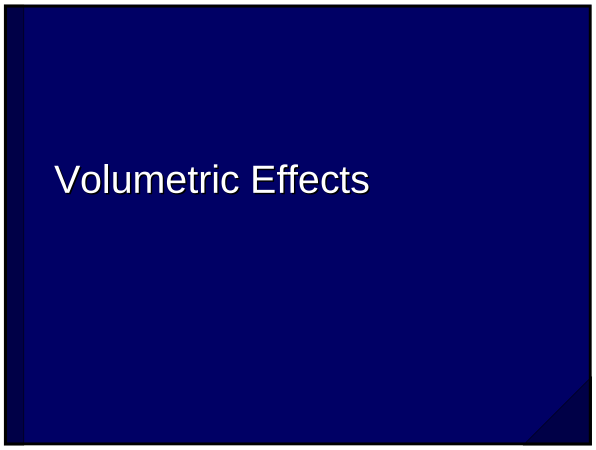


Caustics

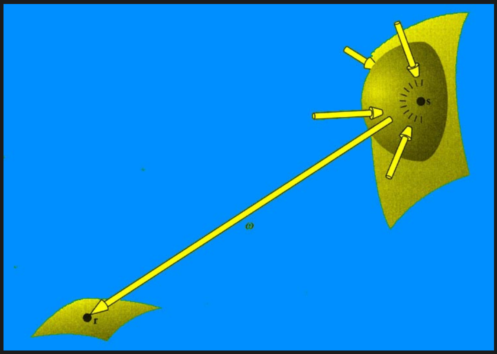
Focused light

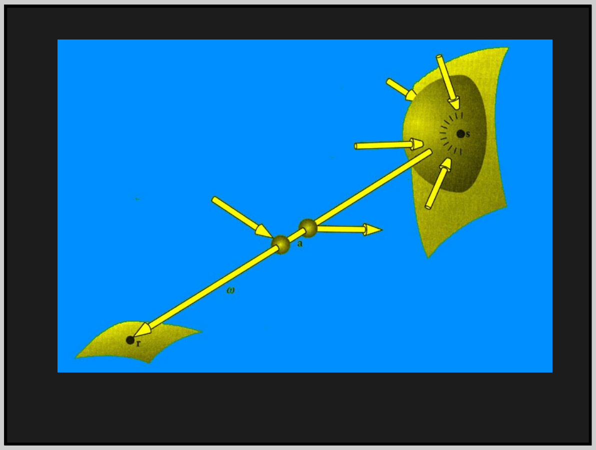


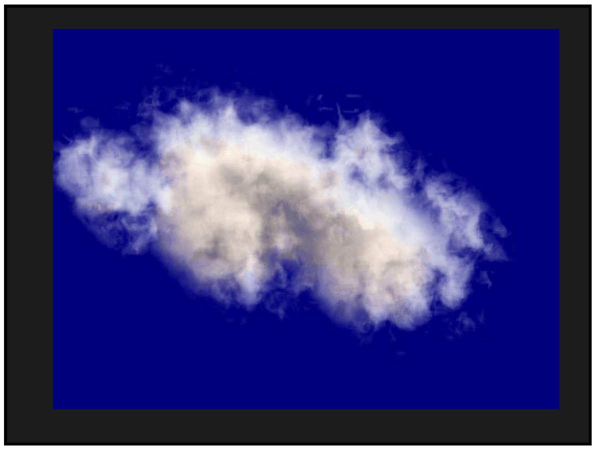




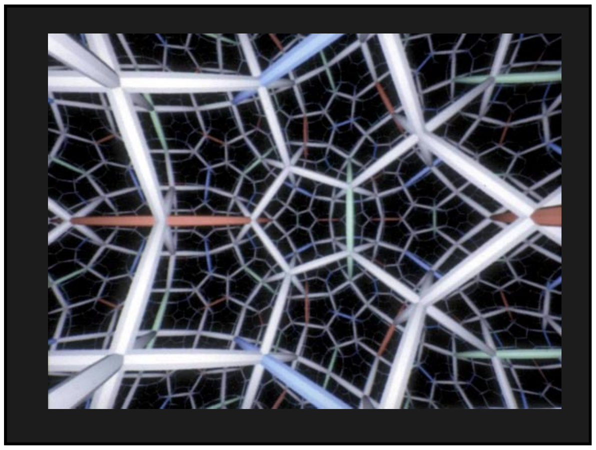
Volumetric Effects

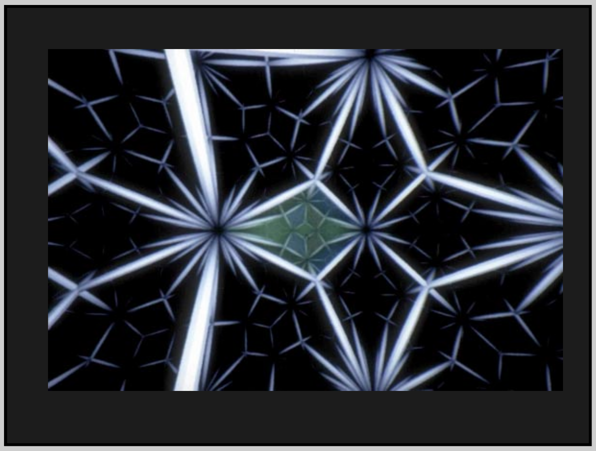


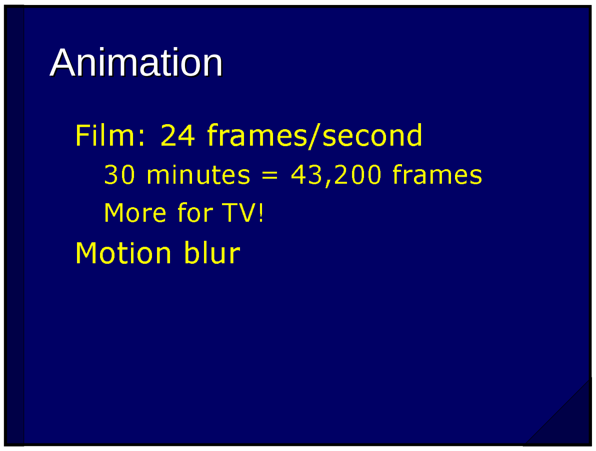




Non-Euclidean Space

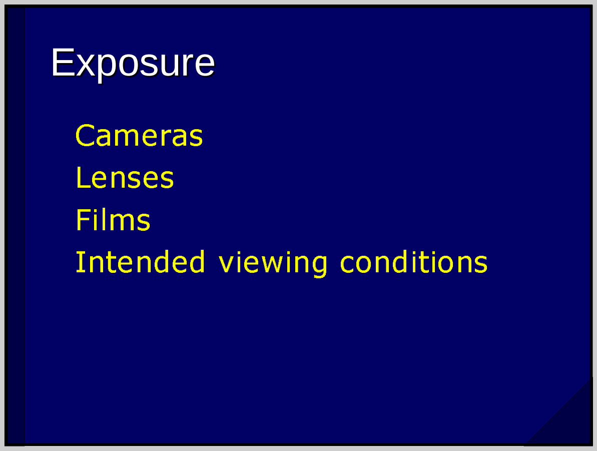






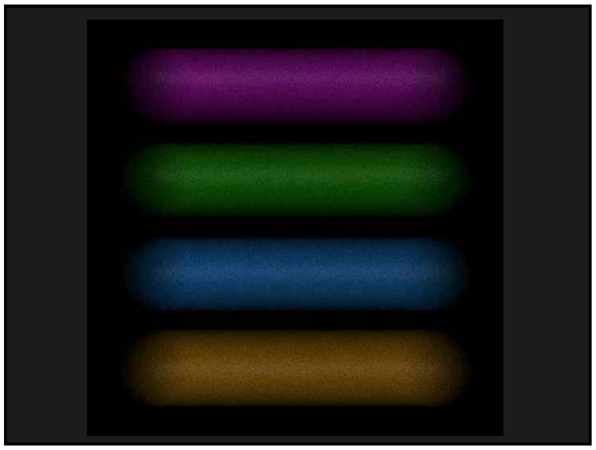
Animation

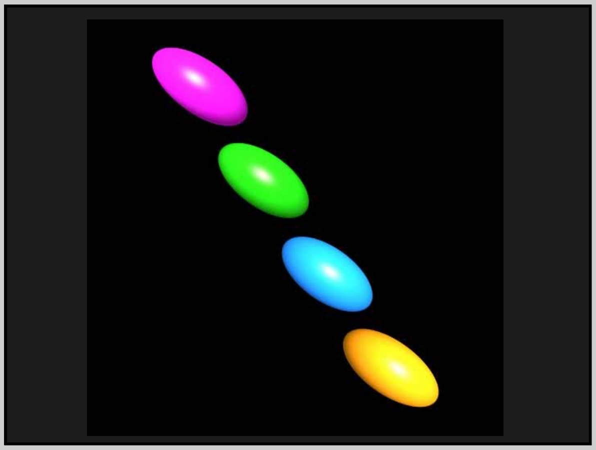
- Film: 24 frames/second
  - 30 minutes = 43,200 frames
  - More for TV!
- Motion blur

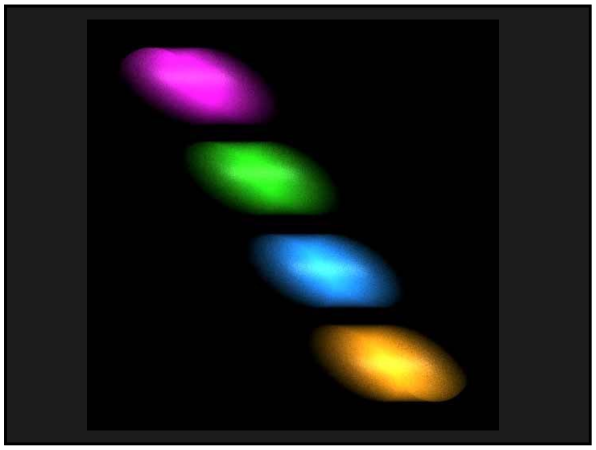


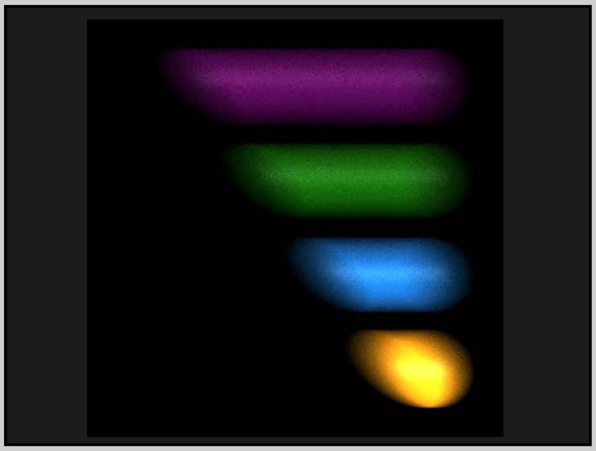
Exposure

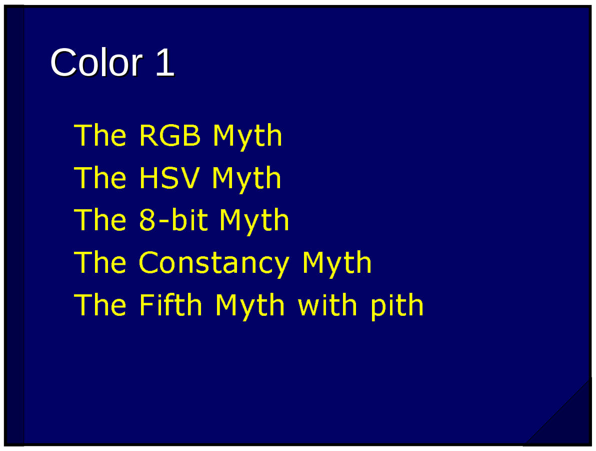
- Cameras
- Lenses
- Films
- Intended viewing conditions





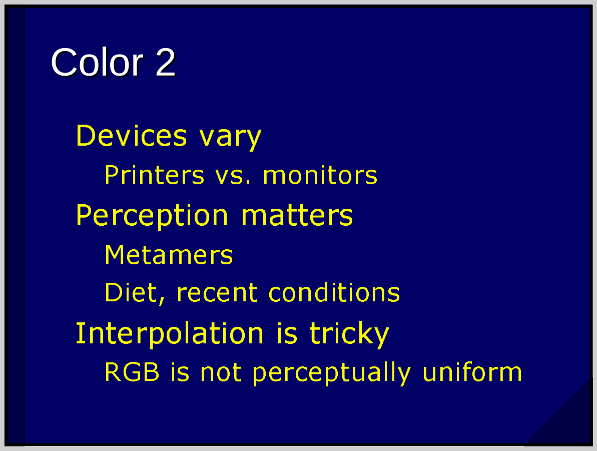






Color 1

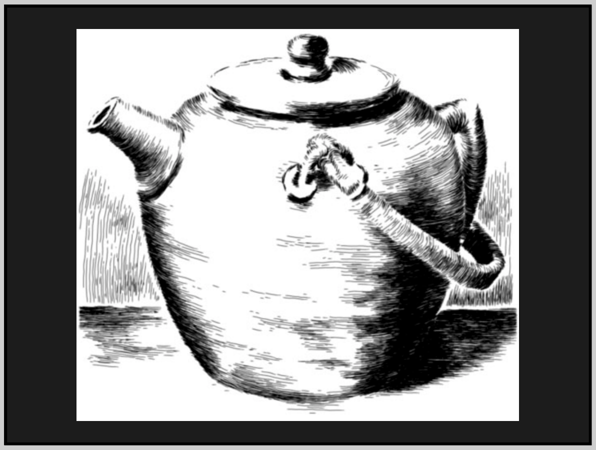
- The RGB Myth
- The HSV Myth
- The 8-bit Myth
- The Constancy Myth
- The Fifth Myth with pith

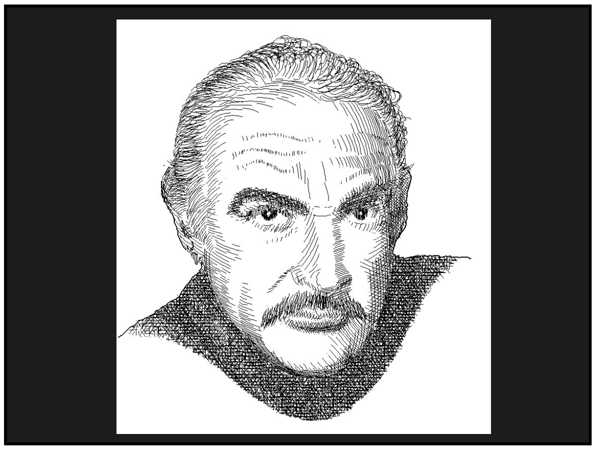


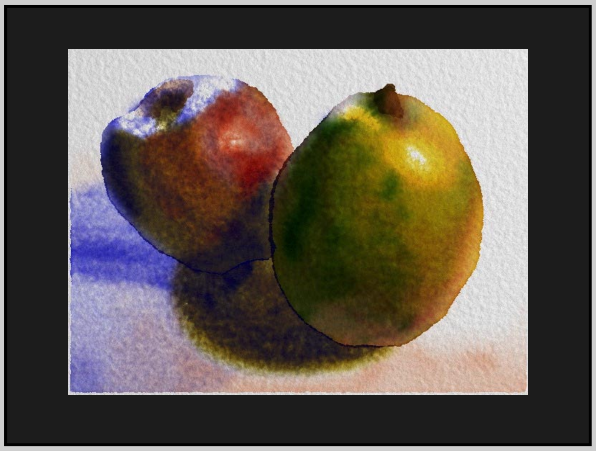
Color 2

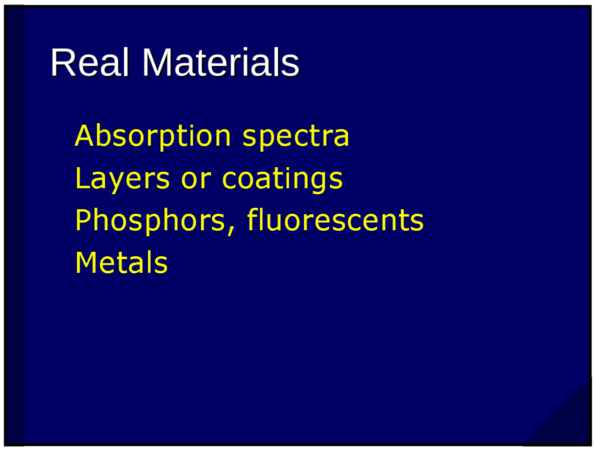
- Devices vary
  - Printers vs. monitors
- Perception matters
  - Metamers
  - Diet, recent conditions
- Interpolation is tricky
  - RGB is not perceptually uniform

Non-Realistic Rendering



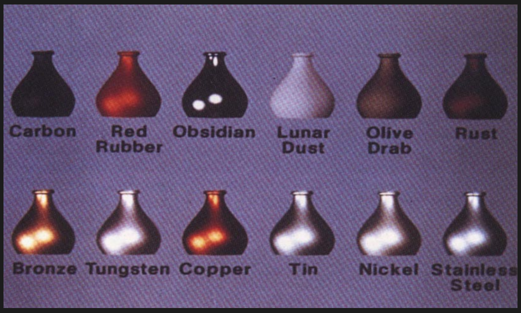


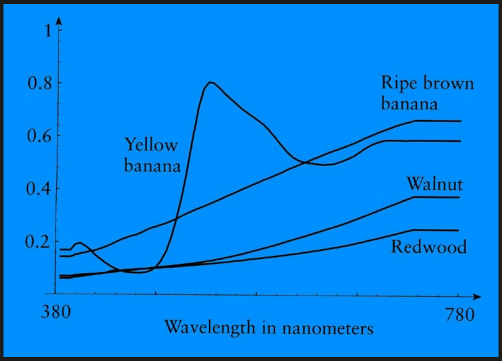


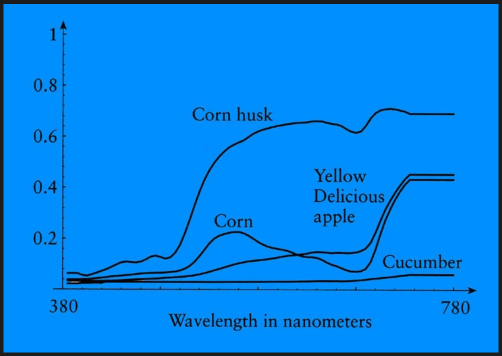


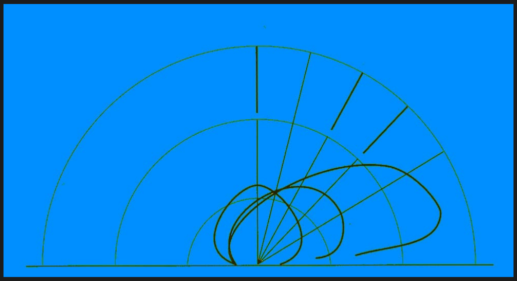
Real Materials

- Absorption spectra
- Layers or coatings
- Phosphors, fluorescents
- Metals









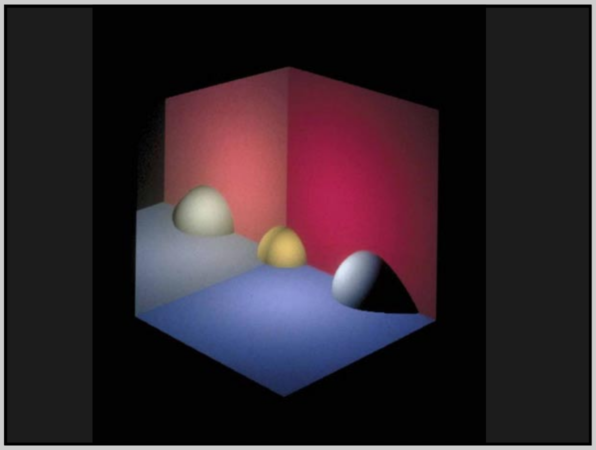
Reflectivity of Paper

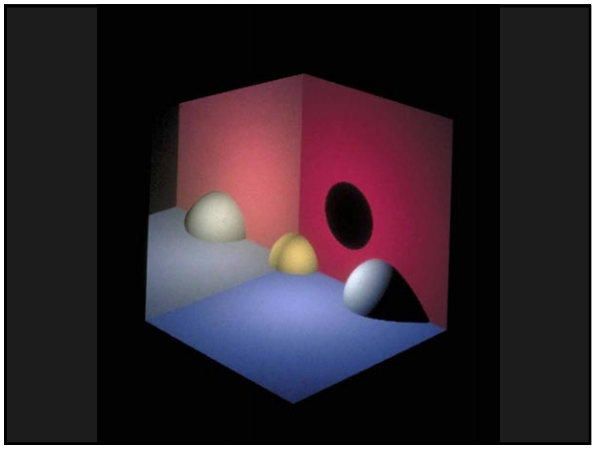
Images 1

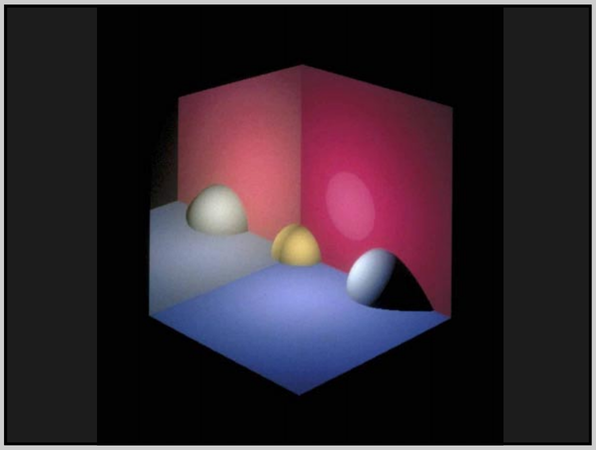
- Frame buffers
  - Pixels
  - 8 bits/color
- Samples
  - 1 sample/pixel
  - Supersampling

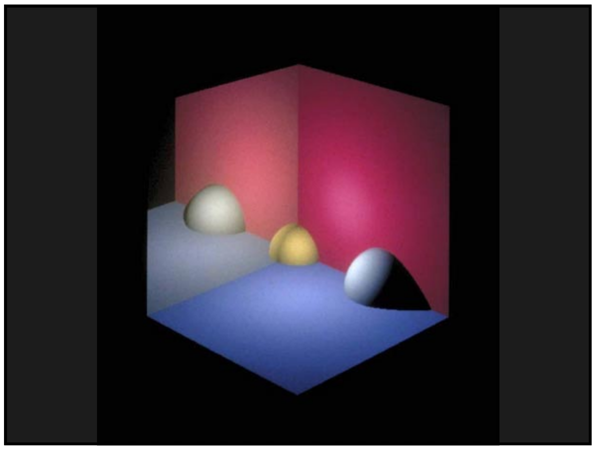
Images 2

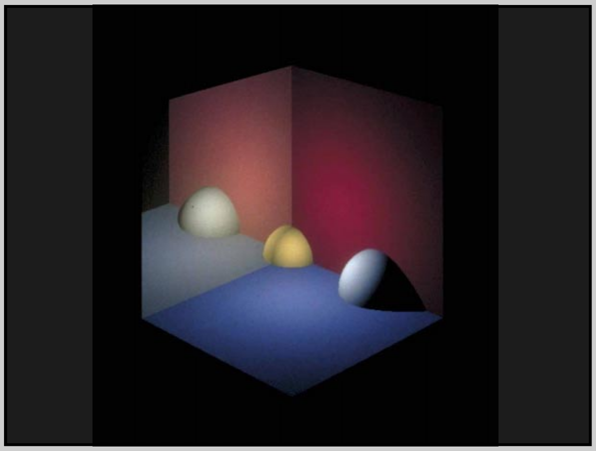
- Storage
  - Components
    - Integer, Floating point
  - Symbolic
- Compression
  - Lossy (MPEG, JPEG)
  - Non-lossy (GIF)
  - Speed vs. space

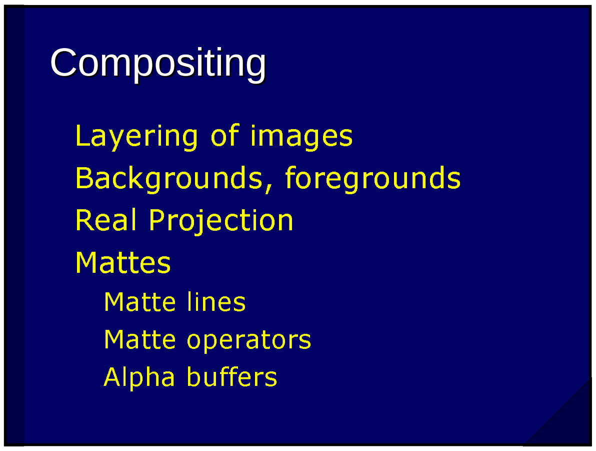






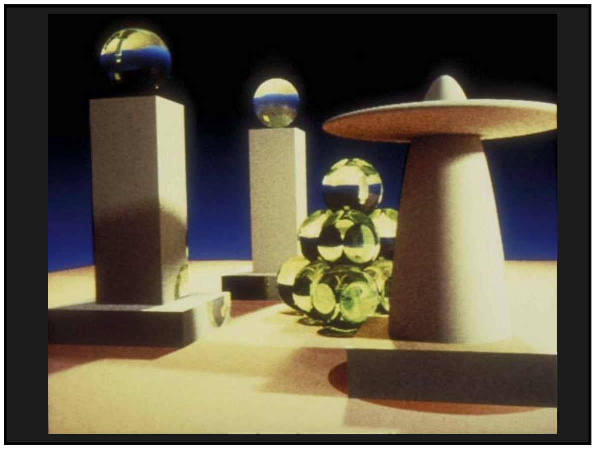


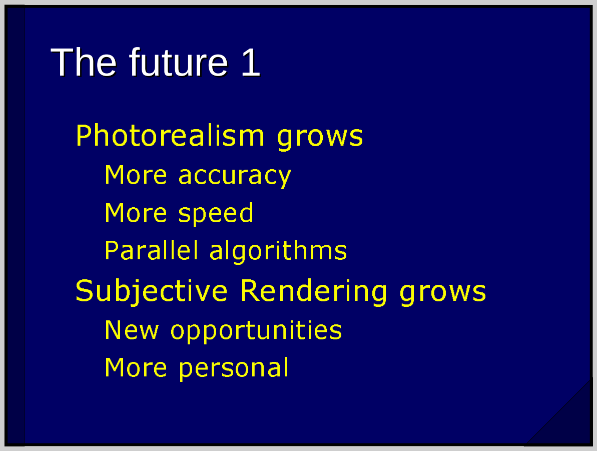




Compositing

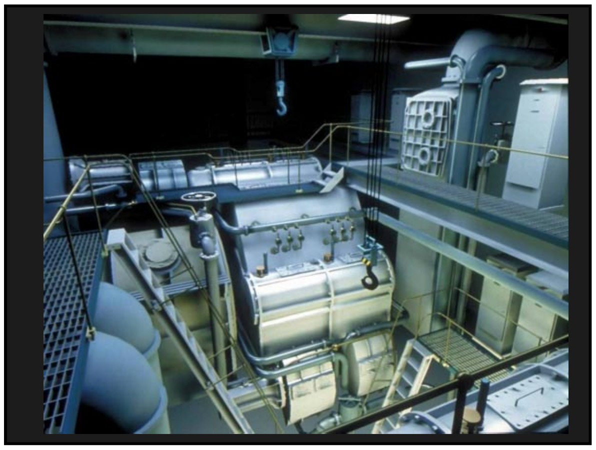
- Layering of images
- Backgrounds, foregrounds
- Real Projection
- Mattes
  - Matte lines
  - Matte operators
  - Alpha buffers

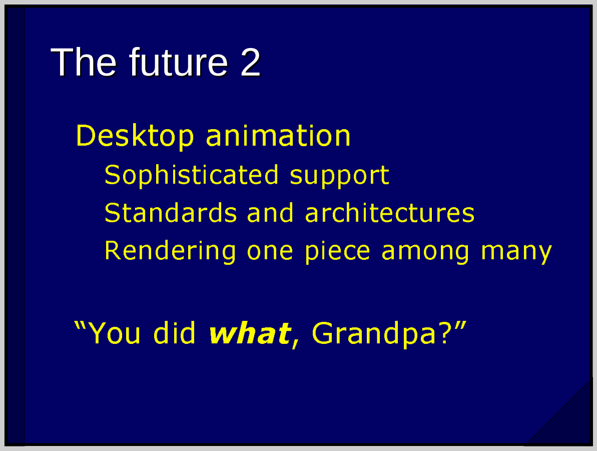




The future 1

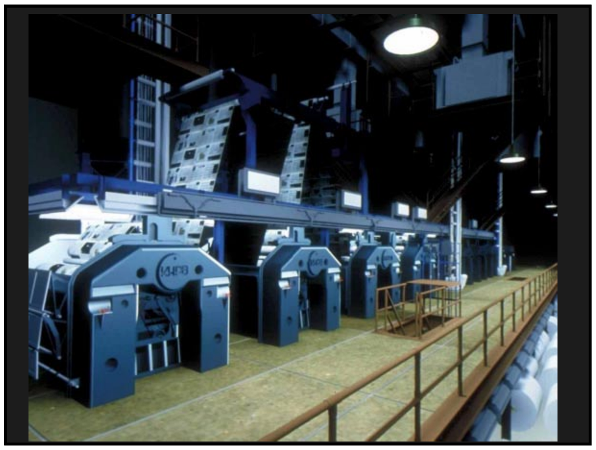
- Photorealism grows
  - More accuracy
  - More speed
  - Parallel algorithms
- Subjective Rendering grows
  - New opportunities
  - More personal

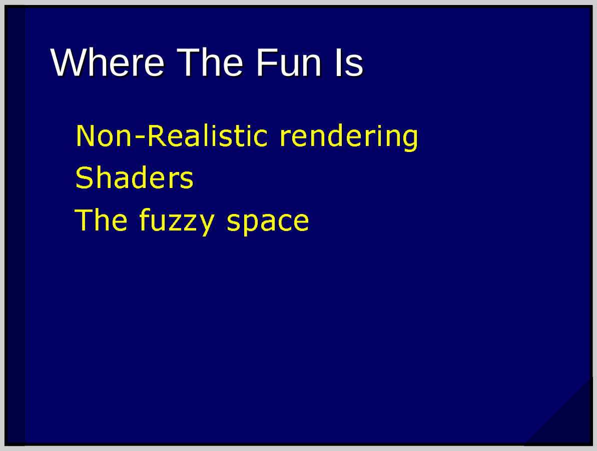




The future 2

- Desktop animation
  - Sophisticated support
  - Standards and architectures
  - Rendering one piece among many
- "You did ***what***, Grandpa?"





Where The Fun Is

- Non-Realistic rendering
- Shaders
- The fuzzy space

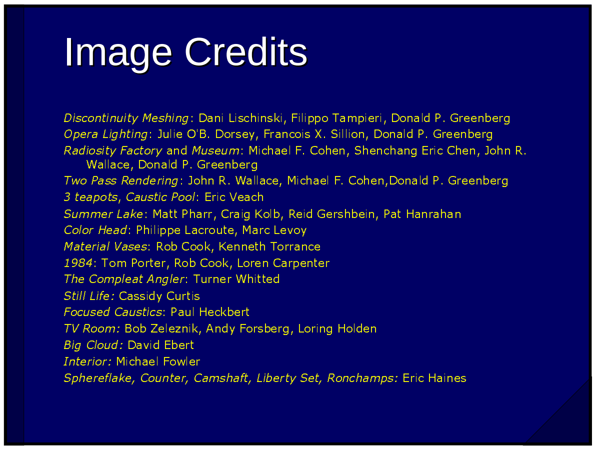
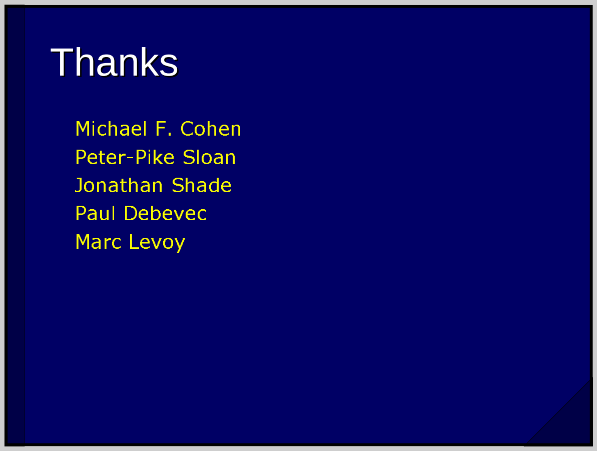


Image Credits

Discontinuity Meshing: Dani Lischinski, Filippo Tampieri, Donald P. Greenberg
Opera Lighting: Julie O'B. Dorsey, Francois X. Sillion, Donald P. Greenberg
Radiosity Factory and *Museum*: Michael F. Cohen, Shenchang Eric Chen, John R. Wallace, Donald P. Greenberg
Two Pass Rendering: John R. Wallace, Michael F. Cohen,Donald P. Greenberg
3 teapots, Caustic Pool: Eric Veach
Summer Lake: Matt Pharr, Craig Kolb, Reid Gershbein, Pat Hanrahan
Color Head: Philippe Lacroute, Marc Levoy
Material Vases: Rob Cook, Kenneth Torrance
1984: Tom Porter, Rob Cook, Loren Carpenter
The Compleat Angler: Turner Whitted
Still Life: Cassidy Curtis
Focused Caustics: Paul Heckbert
TV Room: Bob Zeleznik, Andy Forsberg, Loring Holden
Big Cloud: David Ebert
Interior: Michael Fowler
Sphereflake, Counter, Camshaft, Liberty Set, Ronchamps: Eric Haines



Thanks

- Michael F. Cohen
- Peter-Pike Sloan
- Jonathan Shade
- Paul Debevec
- Marc Levoy

Computer Graphics Hardware

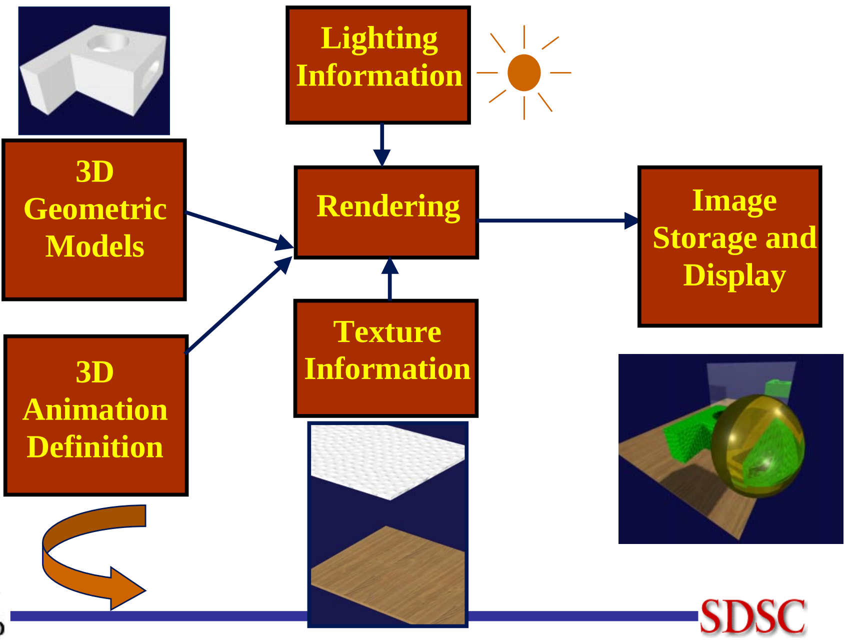
Mike Bailey

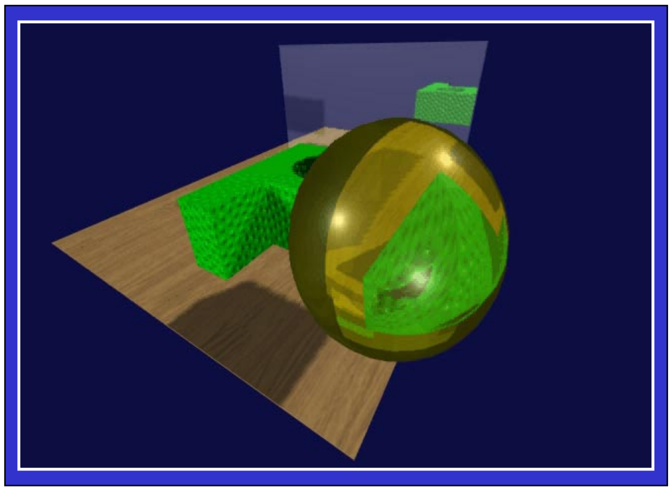
San Diego Supercomputer Center
University of California San Diego

`mjb@sdsc.edu`

`http://www.sdsc.edu/~mjb`

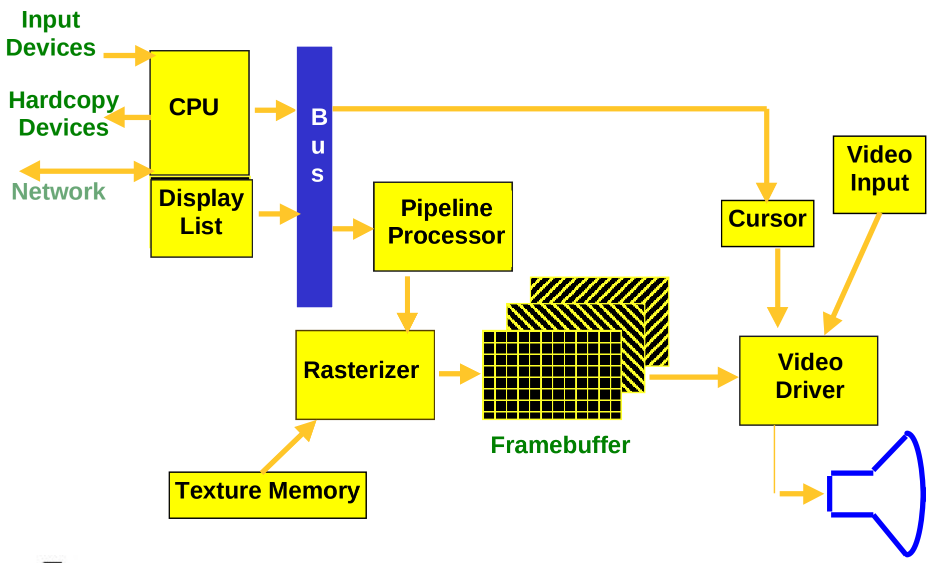
The Generic Graphics Process





UCSD SDSC

The Generic Computer Graphics System

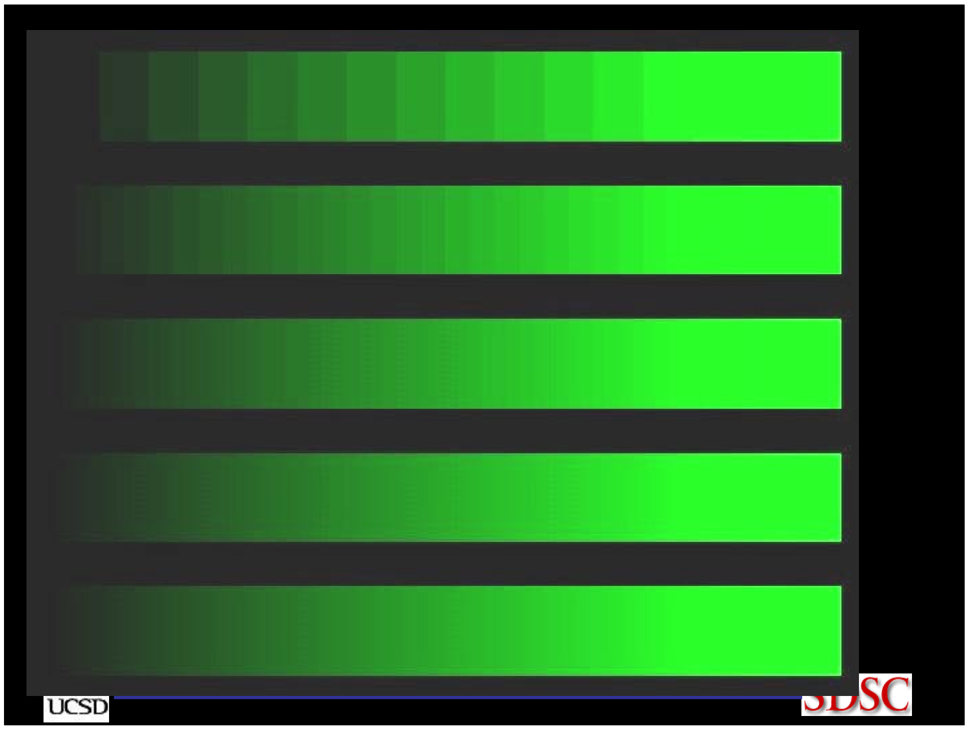


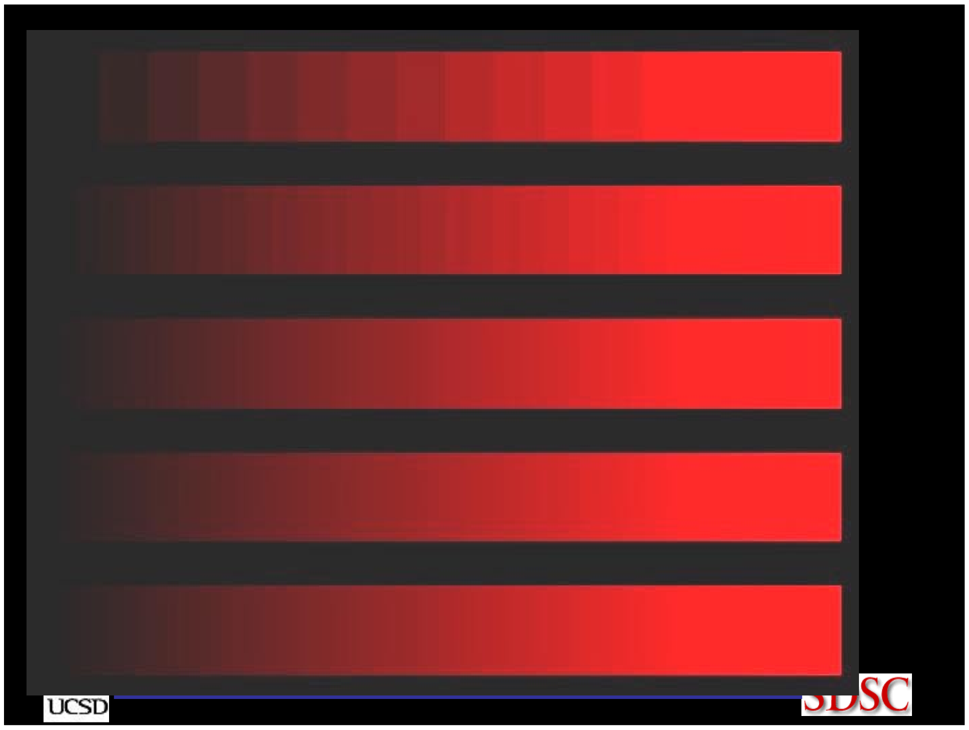
UCSD SDSC

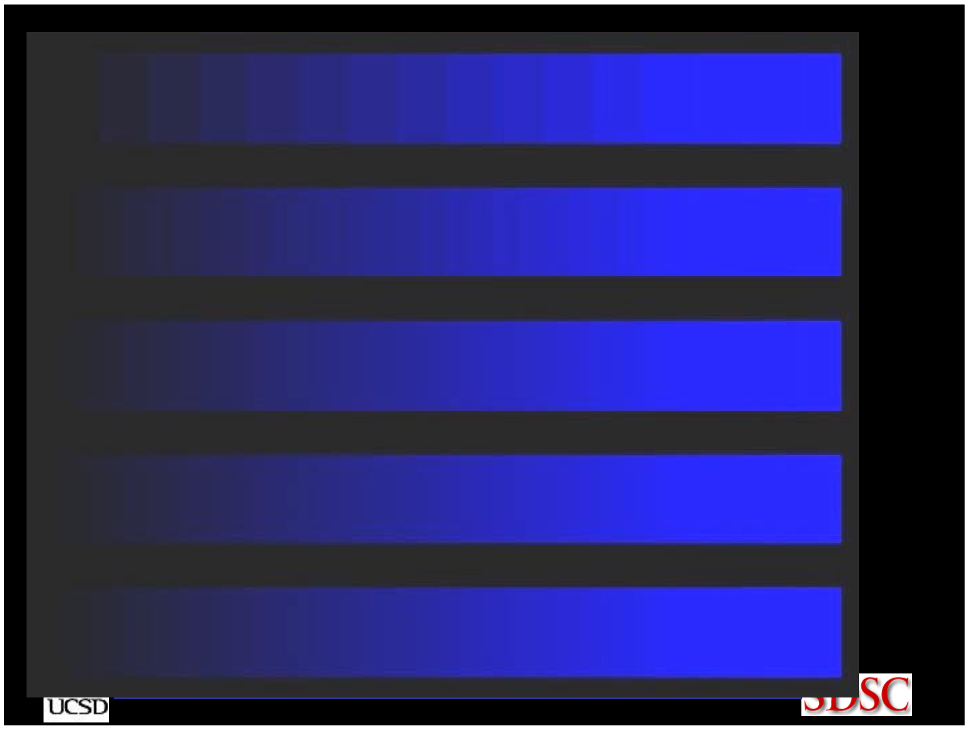
The Human

- ***Acuity:*** **1 arc-minute for those with 20/20 vision**
- **Required *refresh rate*: 40-80 refreshes/second**
- **Required *update rate*: 15+ frames/second**
- **Red, Green, Blue receptors in the eye**

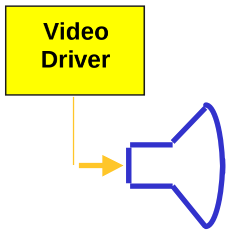
- **How many colors are we able to detect?**




UCSD
SDSC


UCSD
SDSC

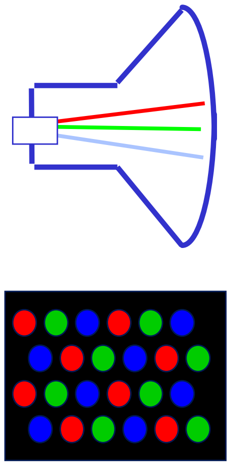
The Computer Graphics Monitor



UCSD SDSC

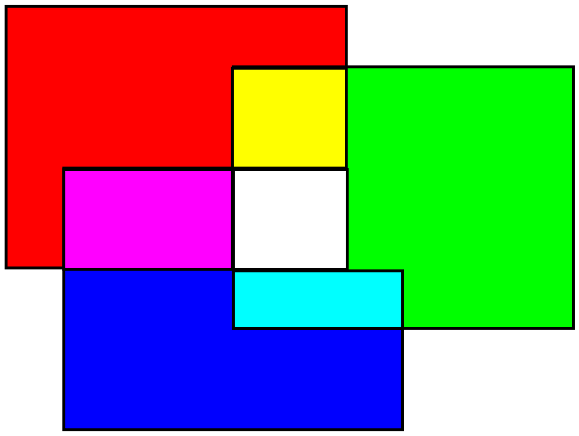
Displaying Color on a Computer Graphics Monitor

- 3 color guns
- Red-green-blue phosphors
- Gun voltage ≈ color brightness



UCSD SDSC

Additive Color (RGB)



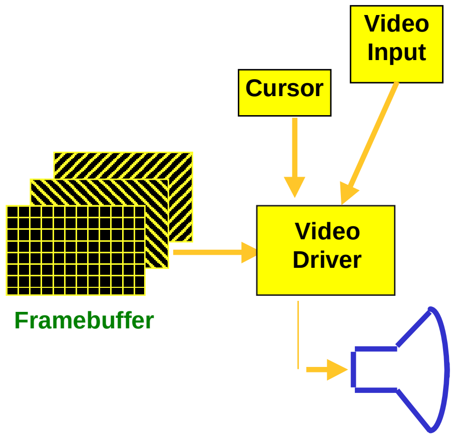
UCSD SDSC

Display Resolution

- *Pixel* resolutions (640x480 - 1600x1024 are common)
- Screen size (13", 16", 19", 21" are common)
- Human acuity: 1 arc-minute is achieved by viewing a 19" monitor with 1280x1024 resolution from a distance of ~40 inches
- FYI: HDTV is talking about resolutions in the 2048x1152 range

UCSD SDSC

The Video Driver



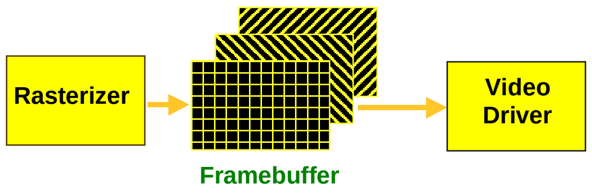
UCSD SDSC

The Video Driver

- N ***refreshes***/second (N is usually between 40 and 80)
- **Framebuffer contains the R,G,B that defines the color at each pixel**
- **Cursor**
  - **Appearance is stored near the video driver in a "mini-framebuffer"**
  - **x,y is given by the CPU**
- **Video input**

UCSD SDSC

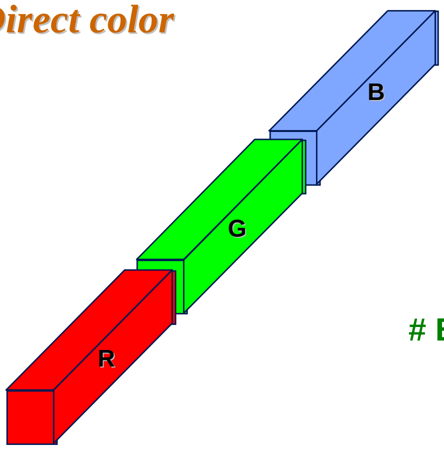
The Framebuffer



UCSD SDSC

The Framebuffer

• *Direct color*



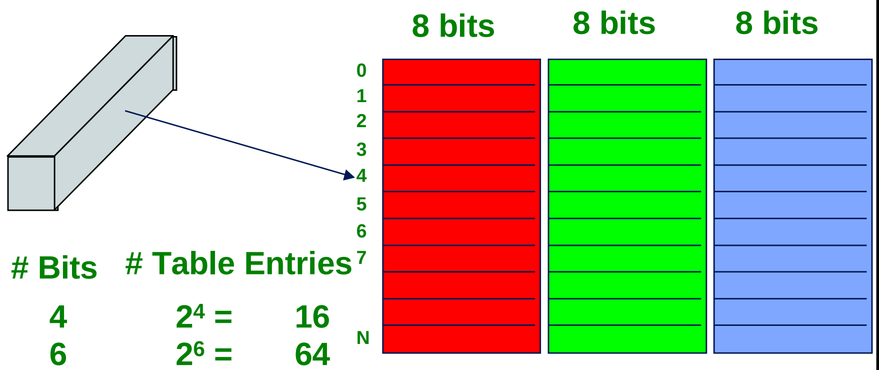
# Bits/pixel	Total colors:
12	2^{12} = 4K
18	2^{18} = 256K
24	2^{24} = 16.7M

# Bits/color	# Shades per color
4	2^4 = 16
6	2^6 = 64
8	2^8 = 256

UCSD SDSC

The Framebuffer

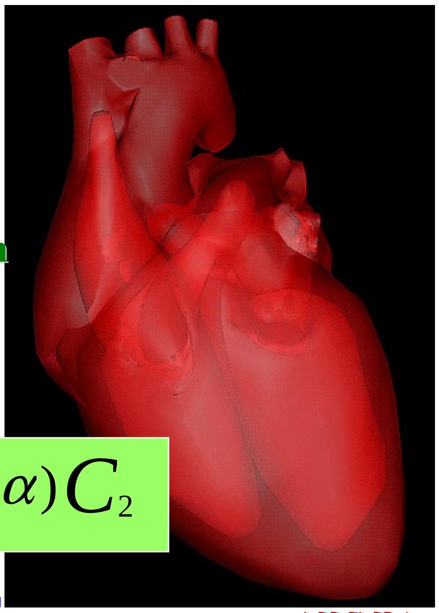
- ***Indirect color*** **with** ***color lookup table***



# Bits	# Table Entries
4	$2^4 =$ 16
6	$2^6 =$ 64
8	$2^8 =$ 256
12	$2^{12} =$ 4096

The Framebuffer

- ***Alpha*** **values**
  - **Transparency per pixel**
    α = 0. is invisible
    α = 1. is opaque
  - **Typically represented in 8 bits**
  - **Alpha blending eqn:**

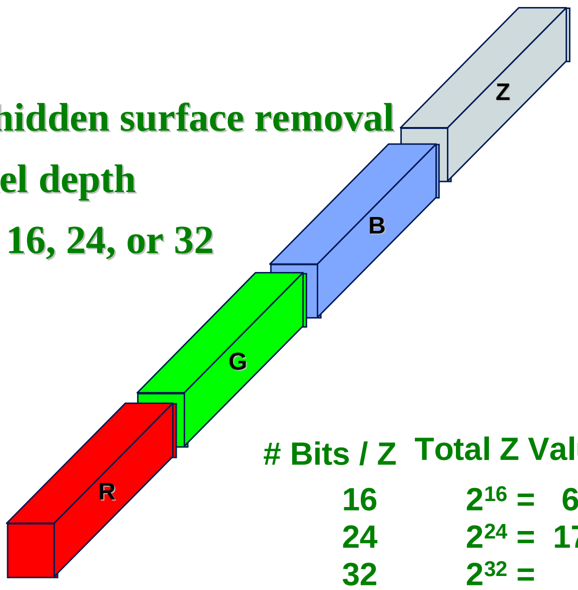


$$Color = \alpha C_1 + (1-\alpha) C_2$$

$0.0 \leq \alpha \leq 1.0$

The Framebuffer

- *Z-buffer*
  - Used for hidden surface removal
  - Holds pixel depth
  - Typically 16, 24, or 32 bits deep



# Bits / Z	Total Z Values:
16	2^{16} = 65K
24	2^{24} = 17M
32	2^{32} = 4B

UCSD SDSC

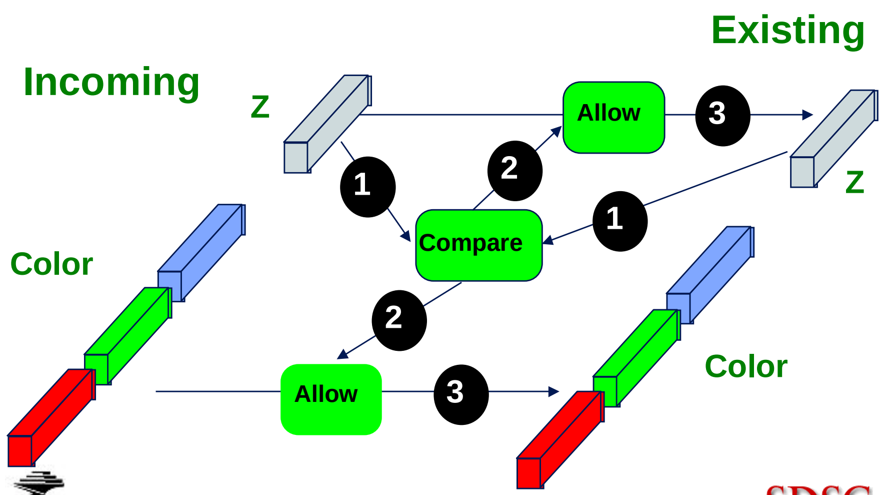
Z-Buffer Steps

1. Compare the incoming Z value with the Z value already in that pixel
2. If the incoming Z value is closer to the viewer than what is already there, open the gates
3. Write the new pixel color and Z value

UCSD SDSC

Z-Buffer Operation

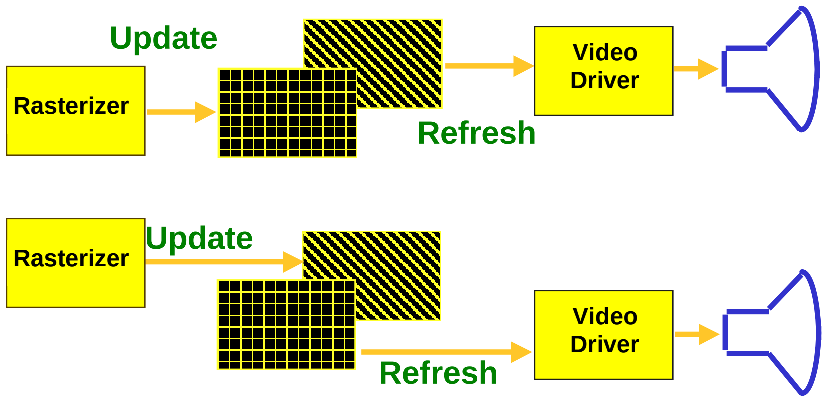
On a pixel-by-pixel basis:



UCSD SDSC

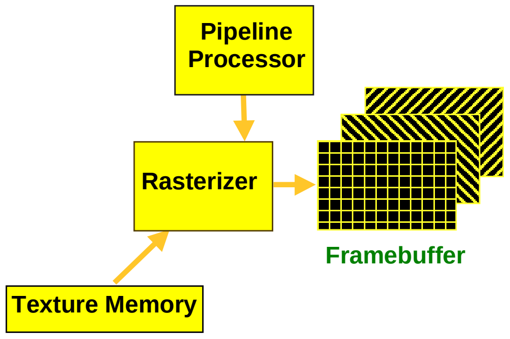
The Framebuffer

Double-buffering: Don’t let the viewer see *any* of the scene until the entire scene is drawn



UCSD SDSC

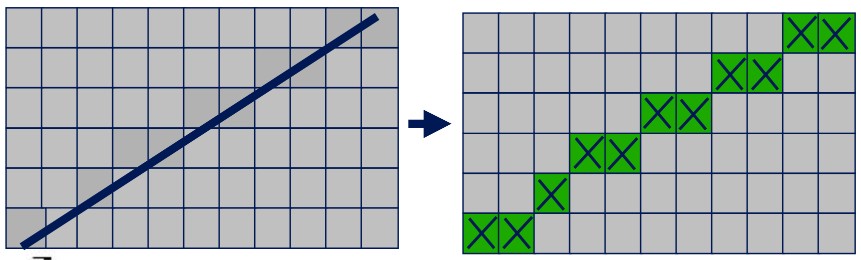
The Rasterizer



UCSD SDSC

Rasterization

- Turn screen space vertex coordinates into pixels that make up lines and polygons
- A great place for custom electronics

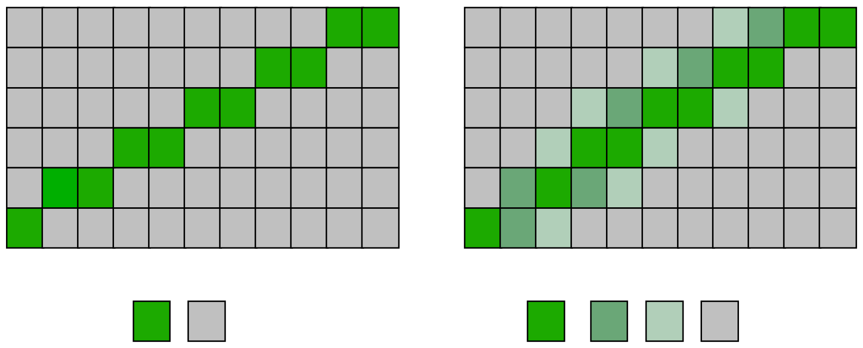


UCSD SDSC

Rasterizers Interpolate:

- X and Y
- Red-green-blue values
- Color index values
- Alpha values
- Z values
- Intensities
- Surface normals
- Texture map coordinates

Anti-Aliasing

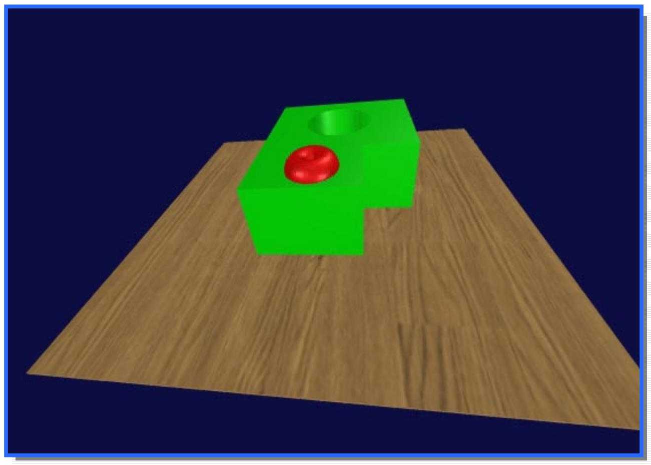


Texture Mapping

- “Stretch” an image onto a piece of geometry
- Image can be generated by a program or scanned in
- Very useful for realistic scene generation

UCSD SDSC

Rasterizers Need To Be Fast

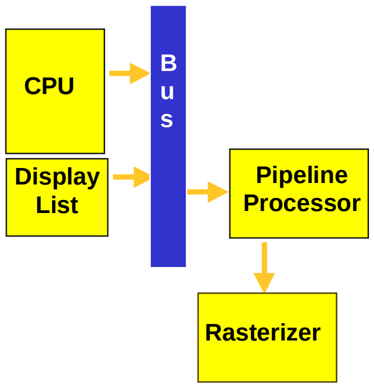


UCSD SDSC

Pipeline Processor

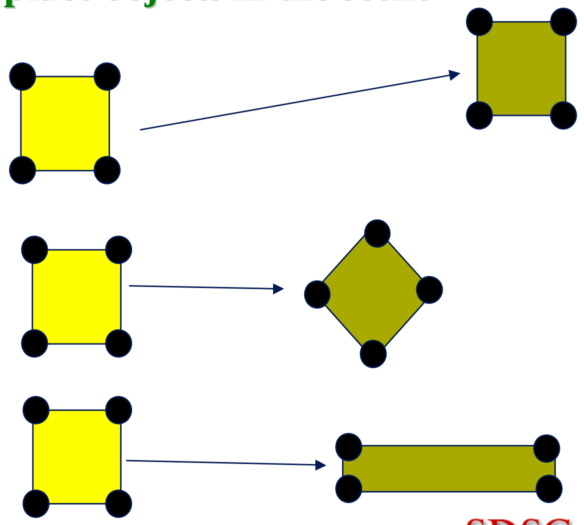
- Coordinates enter in world (application) coordinate space
- Coordinates leave in screen (pixel) coordinate space
- Another great place for custom electronics

The Pipeline Processor



Pipeline Processor: Transformations

- Used to correctly place objects in the scene
- Translation
- Rotation
- Scaling



UCSD SDSC

Pipeline Processor: Windowing and Clipping

- Declare which portion of the 3D universe you are interested in viewing
- This is called the *view volume*
- Clip away everything that is outside the viewing volume

UCSD SDSC

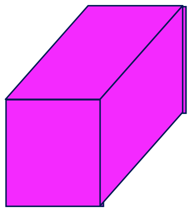
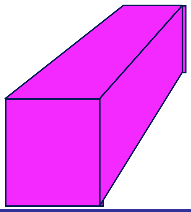
Pipeline Processor: Projection

- Turn 3D coordinates into 2D
  - *Parallel projection*

    Parallel lines remain parallel

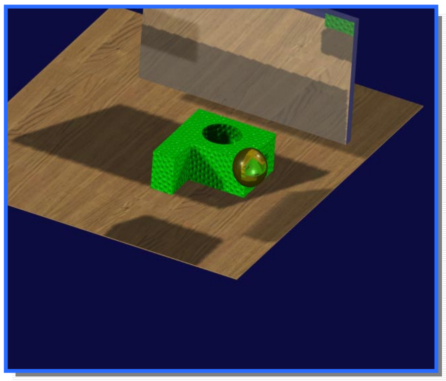
  - *Perspective projection*

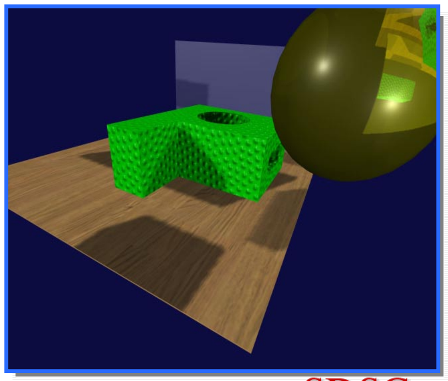
    Some parallel lines appear to converge

UCSD SDSC

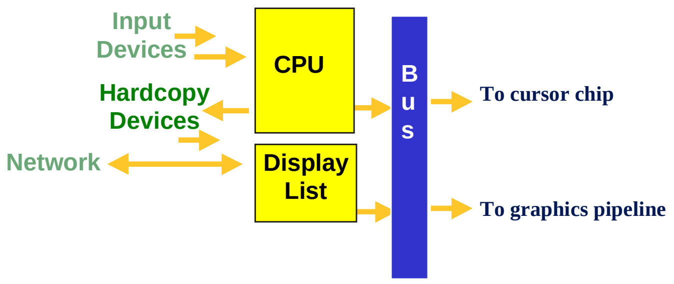
Pipeline Processor: Projection


Parallel

Perspective


UCSD SDSC

The CPU



UCSD SDSC

Display List

- A list of graphics instructions created ahead of time and then "played back" when needed
- May or may not be editable once it is created
- Modern structure of display lists is in the form of a *scene graph*

UCSD SDSC

The Limitations of using NTSC Video

- Cannot display saturated colors well
- Expect an effective resolution of (at best) ~640x480
- Do not use single-pixel thick lines
- Stay away from the edges of the screen
- Some colors have better video resolution than others

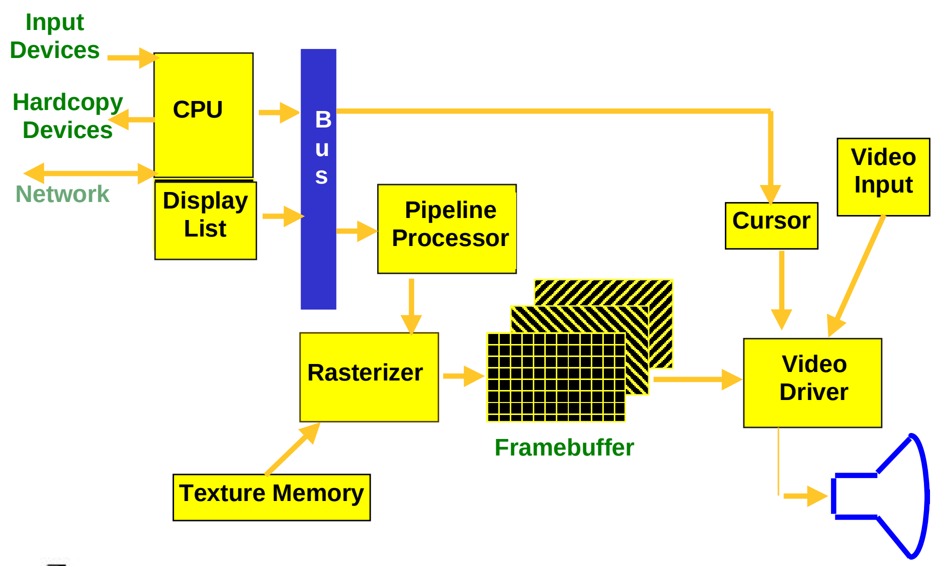
UCSD SDSC

NTSC Cycles of Encoding per Scanline

What:	Cycles/Scanline:
Intensity	267
Orange-Blue	96
Purple-Green	35

UCSD SDSC

All Together, Now!



UCSD SDSC

Graphics Performance

- Because the process has so many steps, it can bottleneck anywhere, depending on the nature of the graphics application
- To evaluate graphics performance, we need a universally-accepted methodology
- Check out the *Graphics Performance Characterization group*:

`http://www.specbench.org/gpc`

UCSD SDSC

Don’t Get Spooked By a Geometric Mean!

$$GM = \sqrt[n]{S_1 \bullet S_2 \bullet S_3 \bullet \ldots \bullet S_n}$$

Scores = 1, 2, 3	Scores = 1, 2, 3, 10
Arithmetic mean = 2.0	Arithmetic mean = 4.0
Geometric mean = 1.8	Geometric mean = 2.8

UCSD SDSC

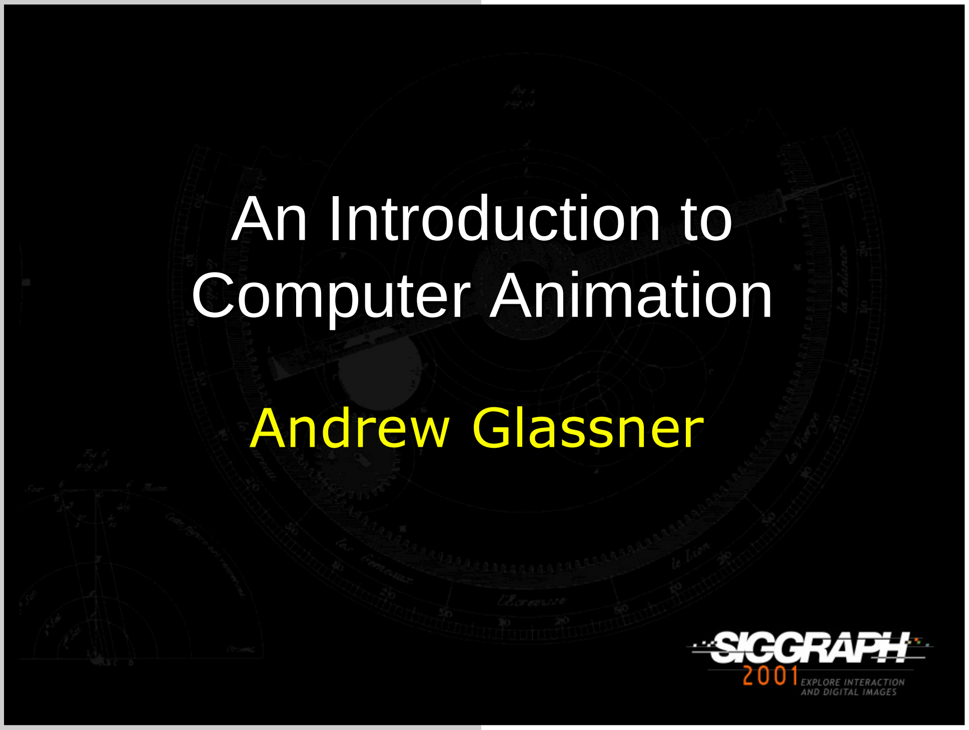
Have Fun, and Thanks for Coming!

Computer Graphics Hardware

Mike Bailey

mjb@sdsc.edu

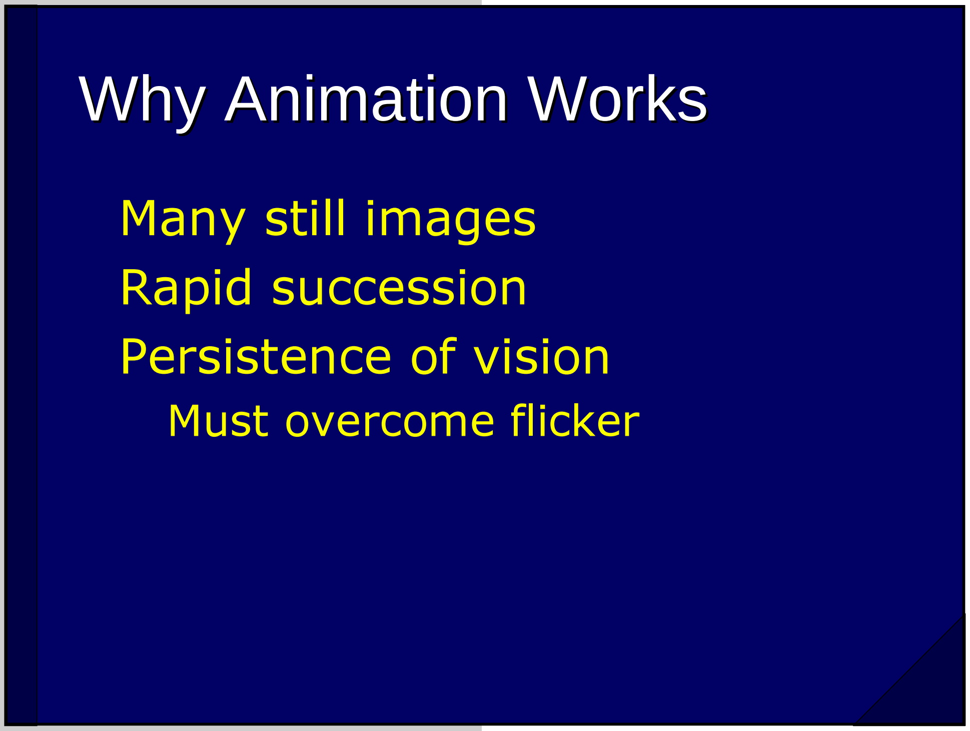
UCSD SDSC



An Introduction to Computer Animation

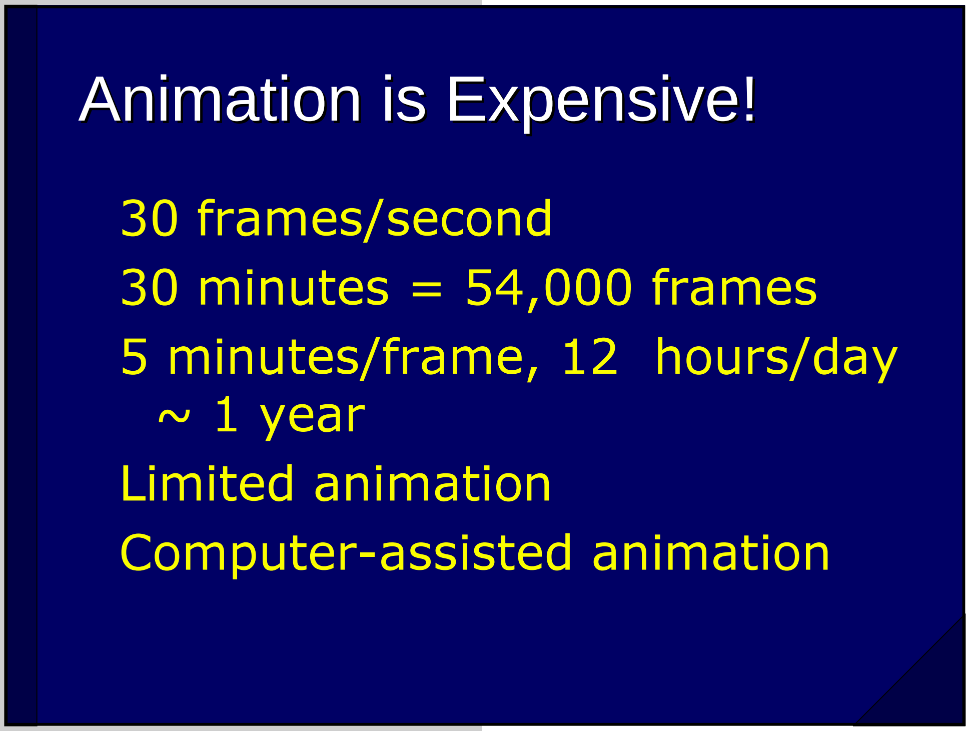
Andrew Glassner

SIGGRAPH 2001 EXPLORE INTERACTION AND DIGITAL IMAGES



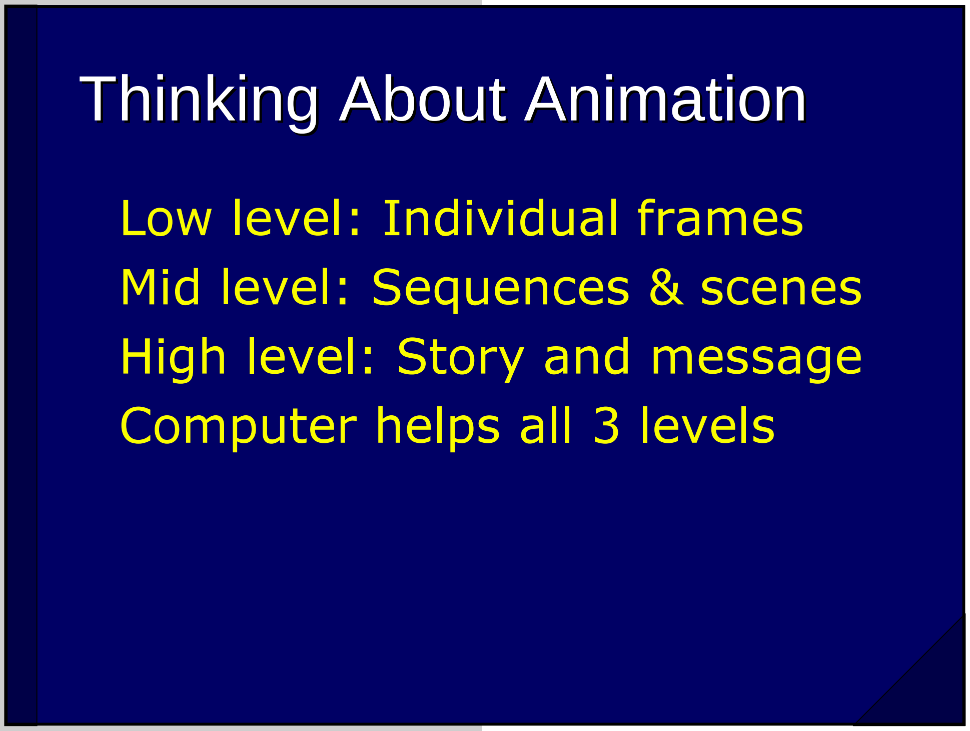
Why Animation Works

- Many still images
- Rapid succession
- Persistence of vision
  - Must overcome flicker



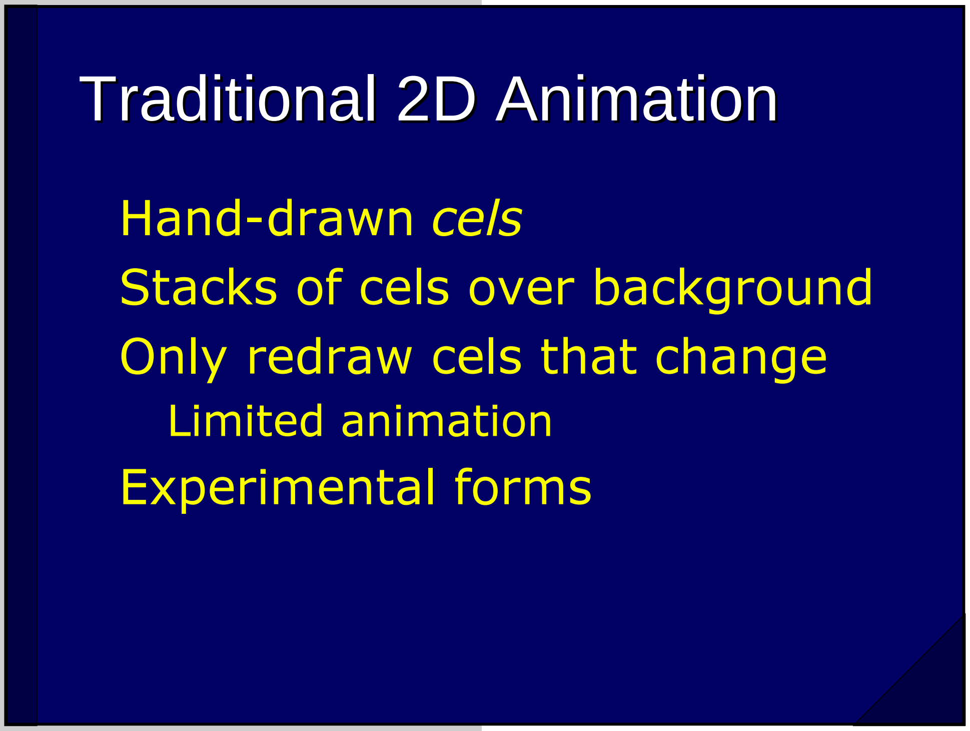
Animation is Expensive!

- 30 frames/second
- 30 minutes = 54,000 frames
- 5 minutes/frame, 12 hours/day
  ~ 1 year
- Limited animation
- Computer-assisted animation



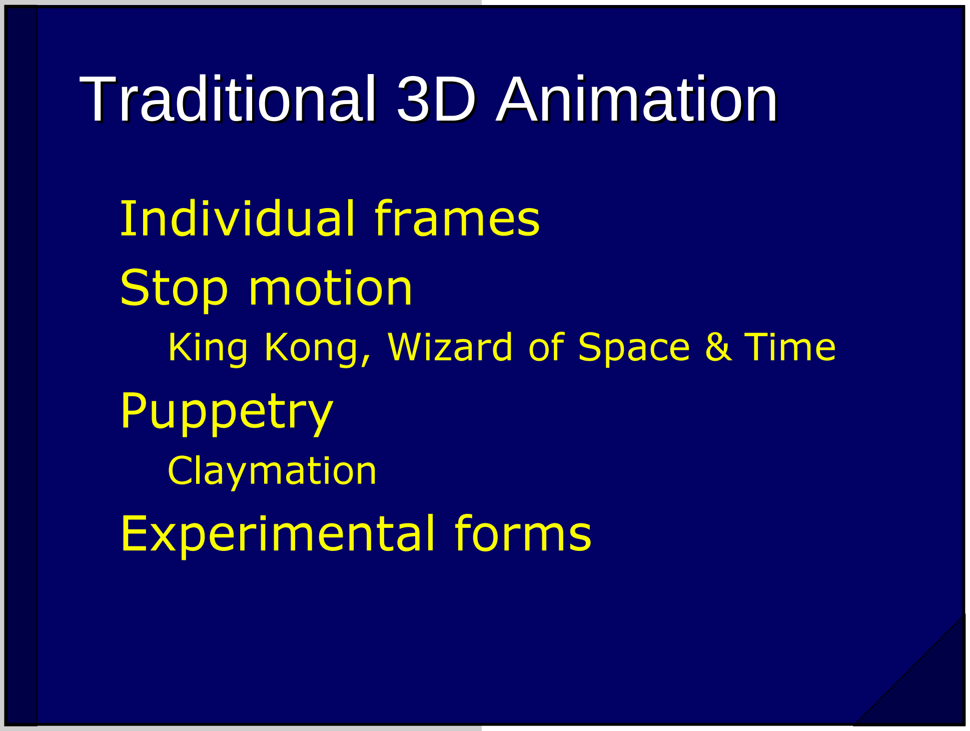
Thinking About Animation

- Low level: Individual frames
- Mid level: Sequences & scenes
- High level: Story and message
- Computer helps all 3 levels



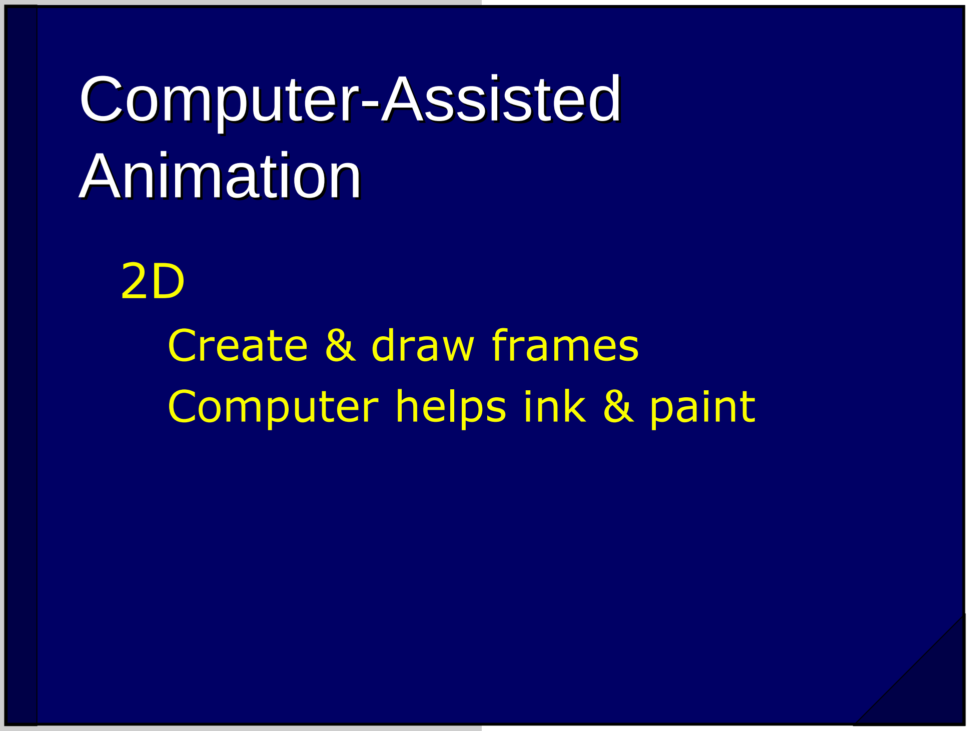
Traditional 2D Animation

- Hand-drawn *cels*
- Stacks of cels over background
- Only redraw cels that change
  - Limited animation
- Experimental forms



Traditional 3D Animation

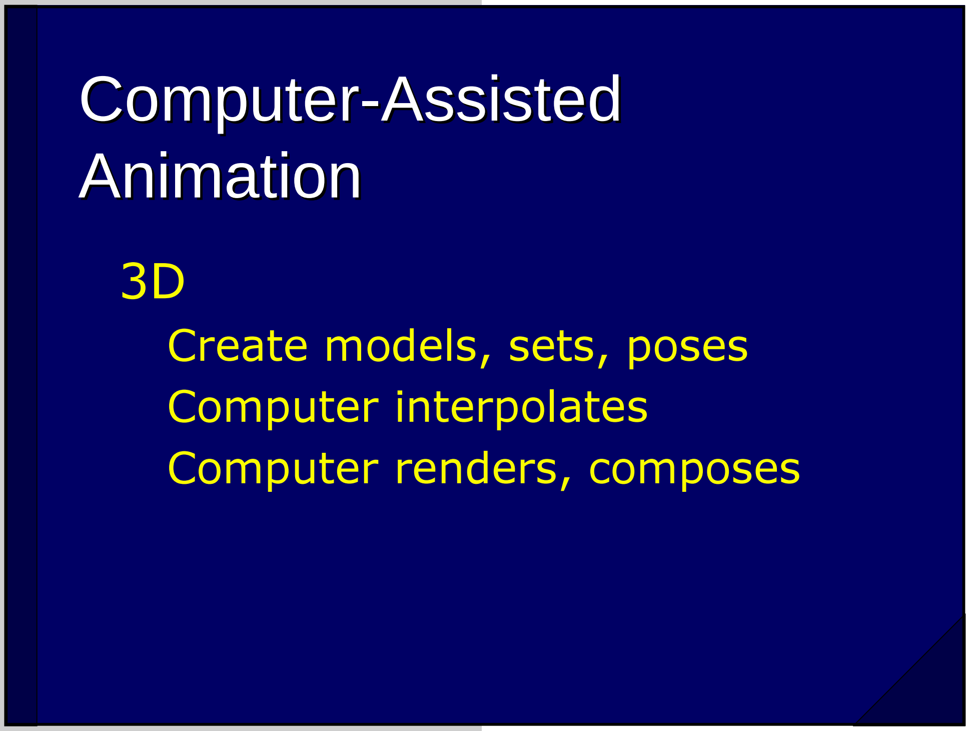
- Individual frames
- Stop motion
  - King Kong, Wizard of Space & Time
- Puppetry
  - Claymation
- Experimental forms



Computer-Assisted Animation

2D

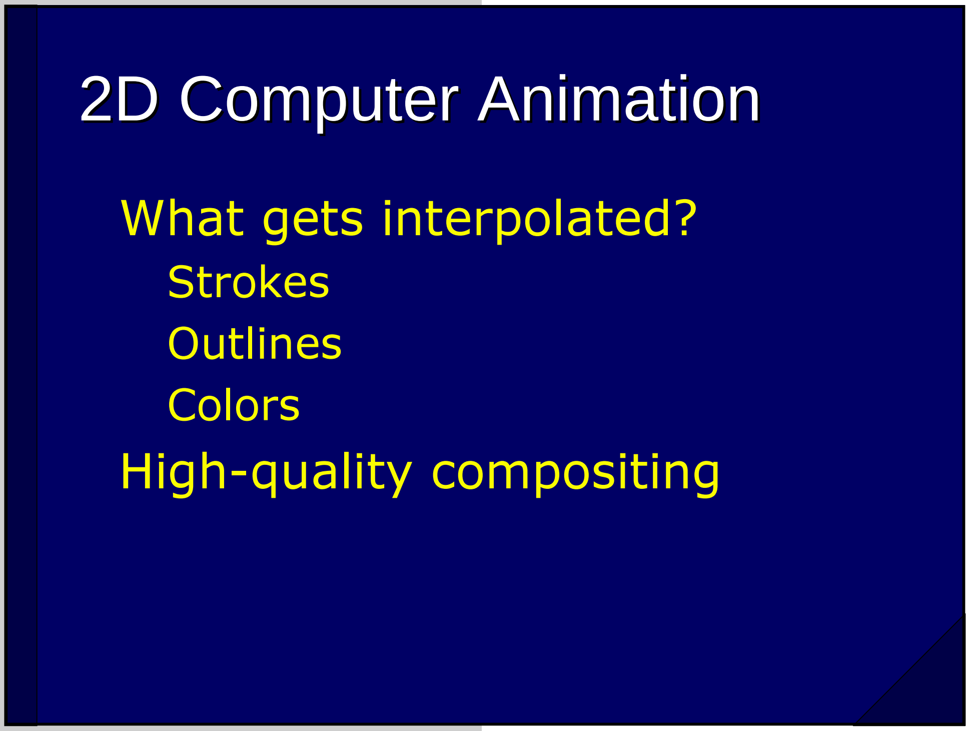
- Create & draw frames
- Computer helps ink & paint



Computer-Assisted Animation

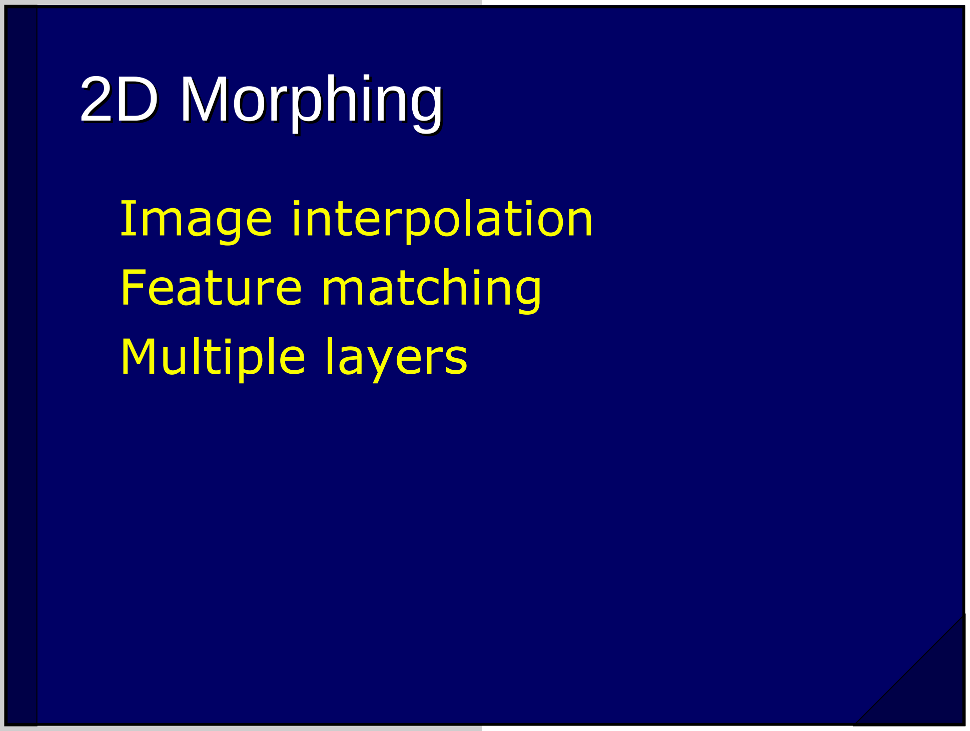
3D

- Create models, sets, poses
- Computer interpolates
- Computer renders, composes



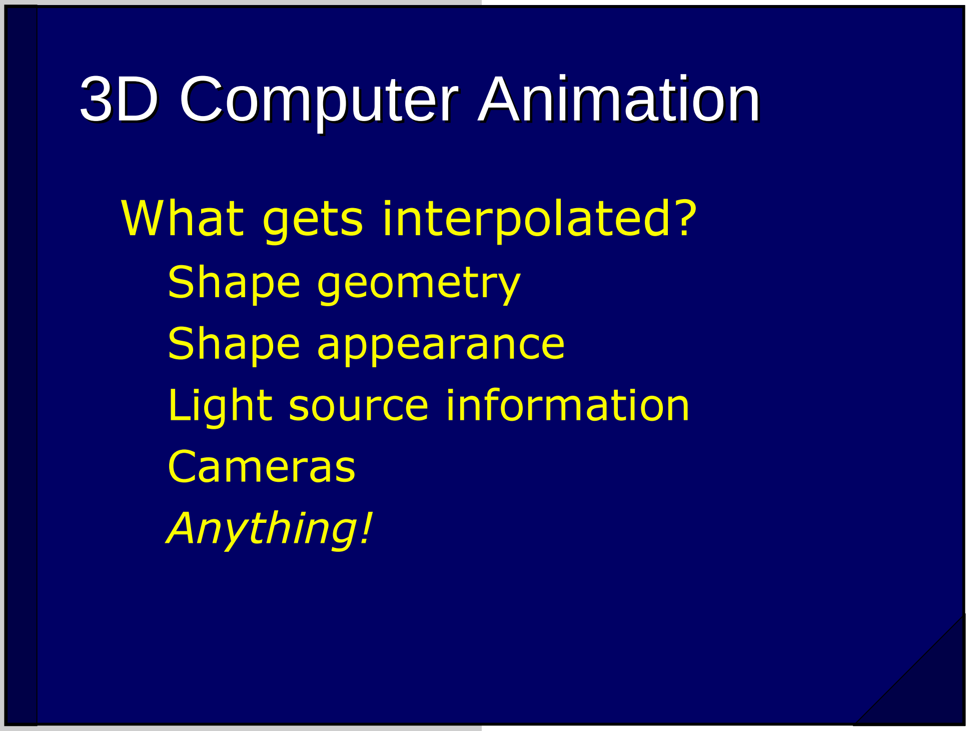
2D Computer Animation

- What gets interpolated?
  - Strokes
  - Outlines
  - Colors
- High-quality compositing



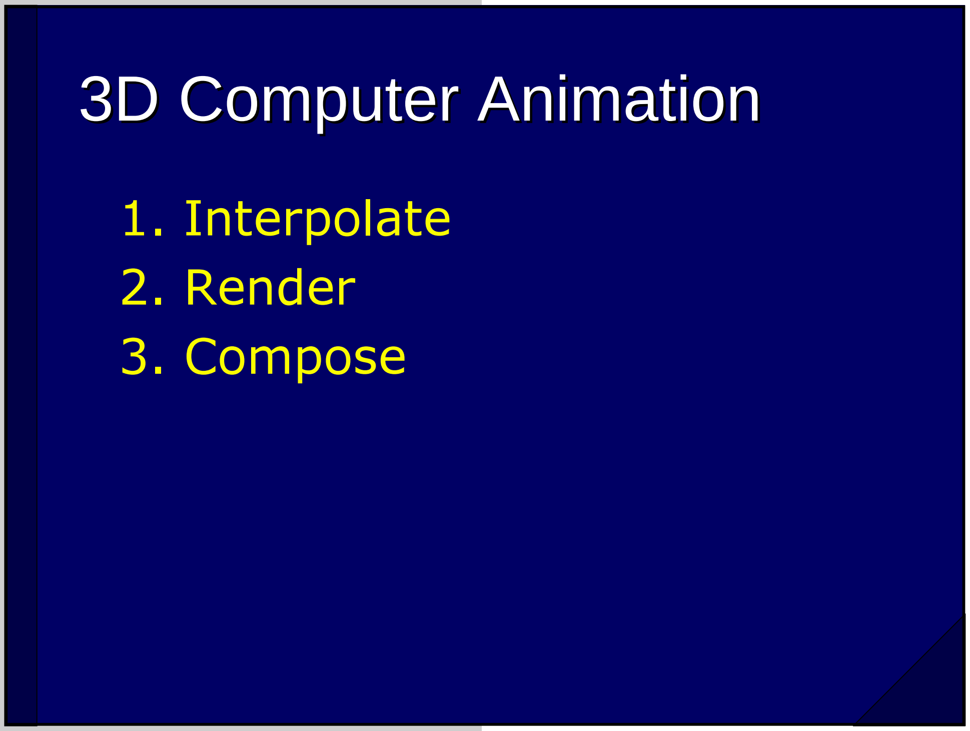
2D Morphing

- Image interpolation
- Feature matching
- Multiple layers



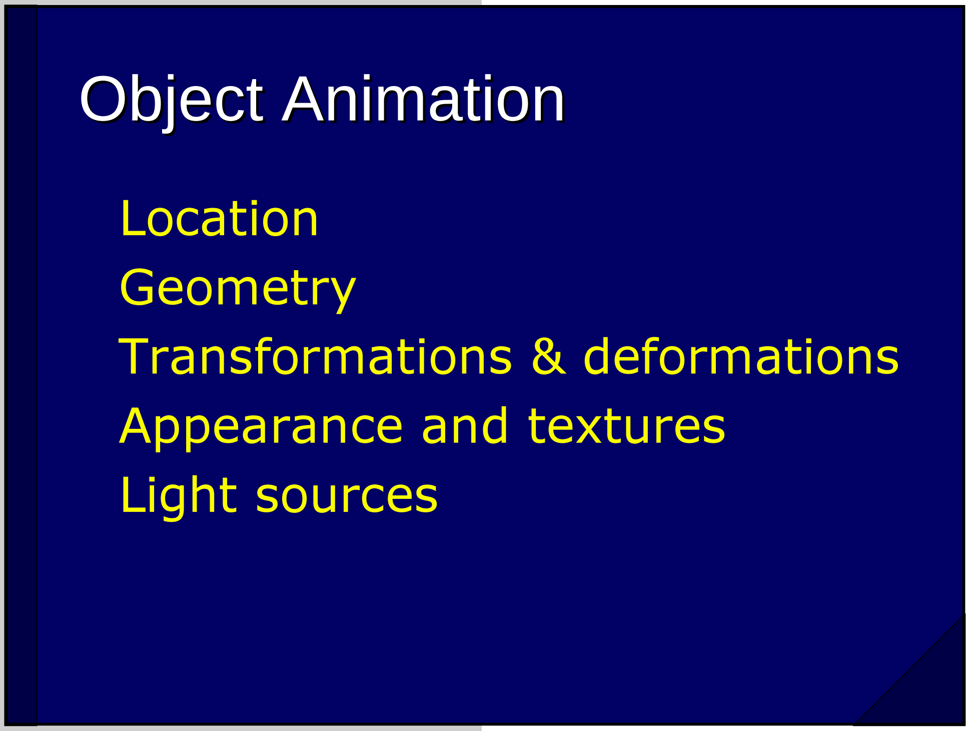
3D Computer Animation

What gets interpolated?
- Shape geometry
- Shape appearance
- Light source information
- Cameras
- *Anything!*



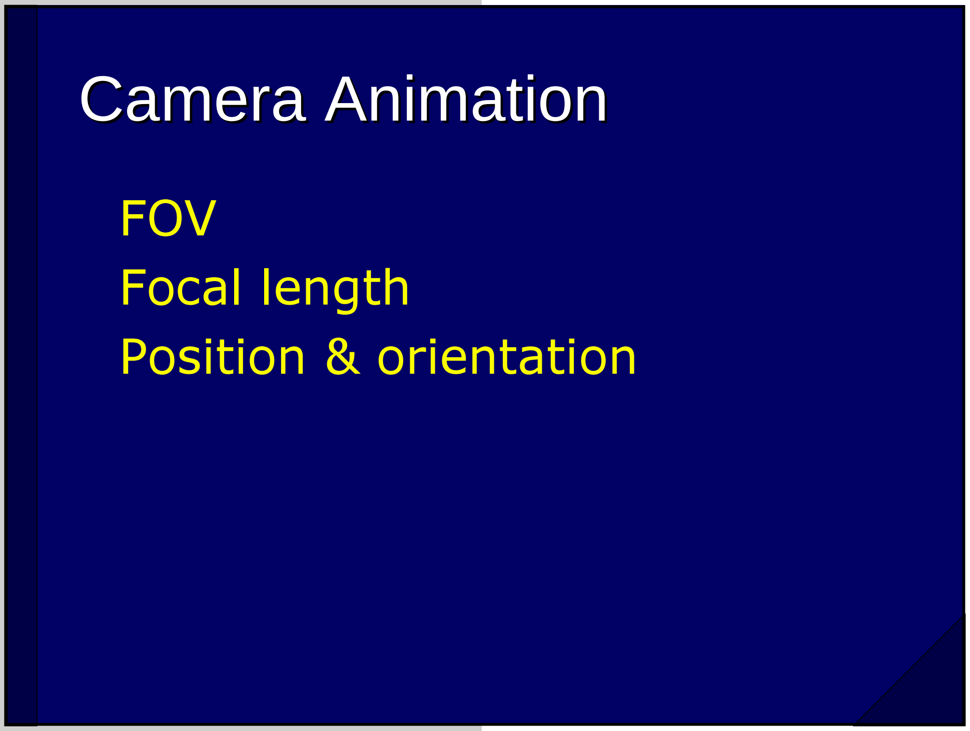
3D Computer Animation

1. Interpolate
2. Render
3. Compose



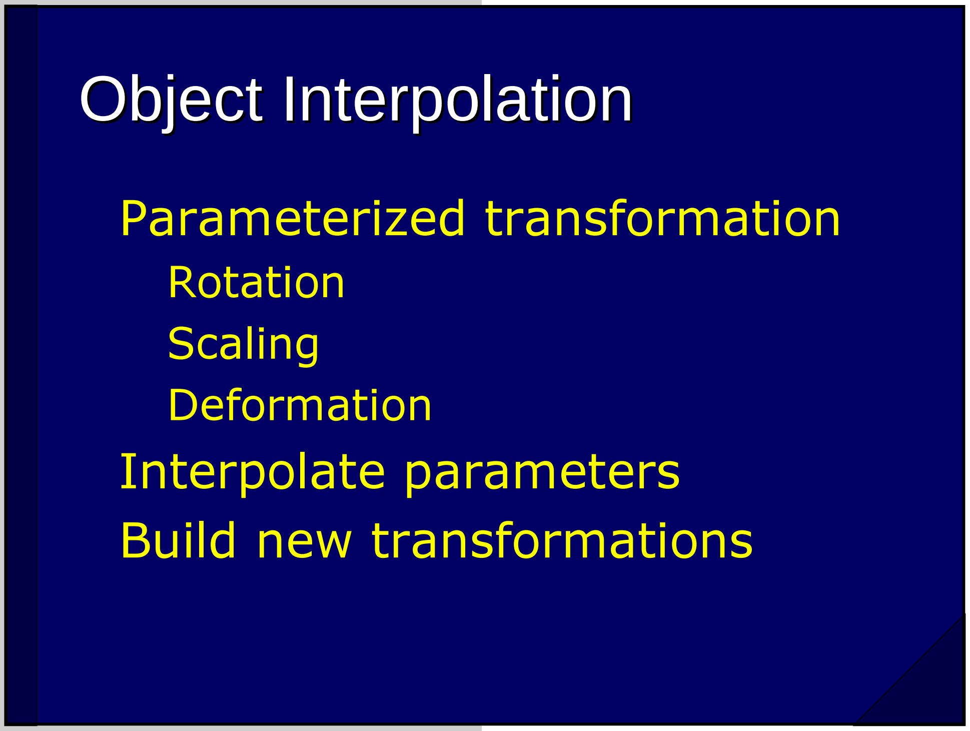
Object Animation

- Location
- Geometry
- Transformations & deformations
- Appearance and textures
- Light sources



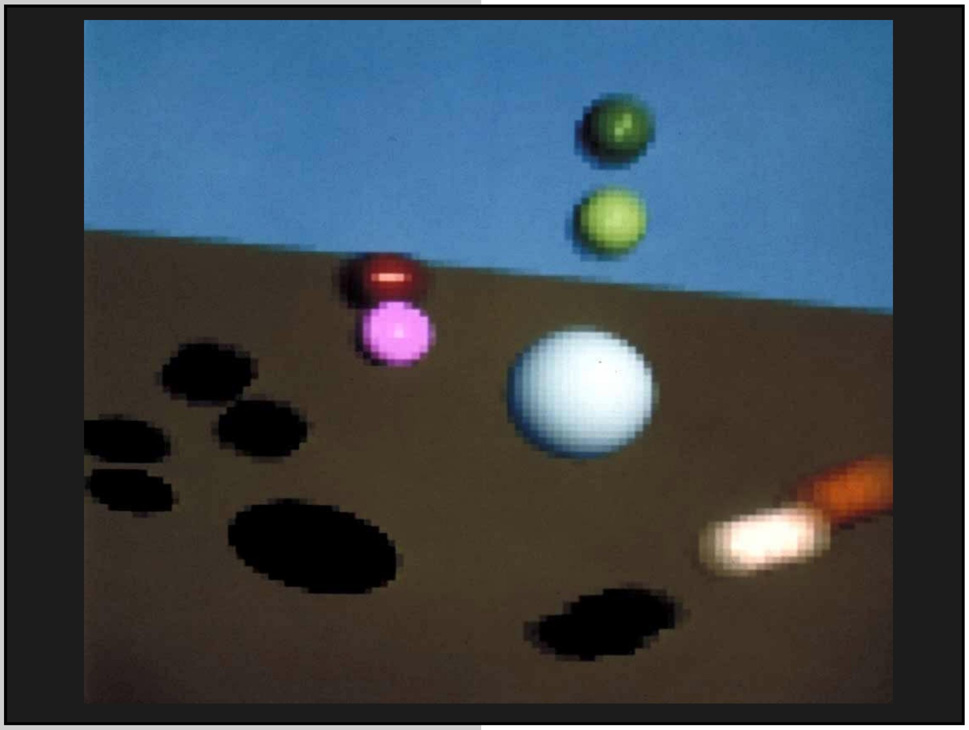
Camera Animation

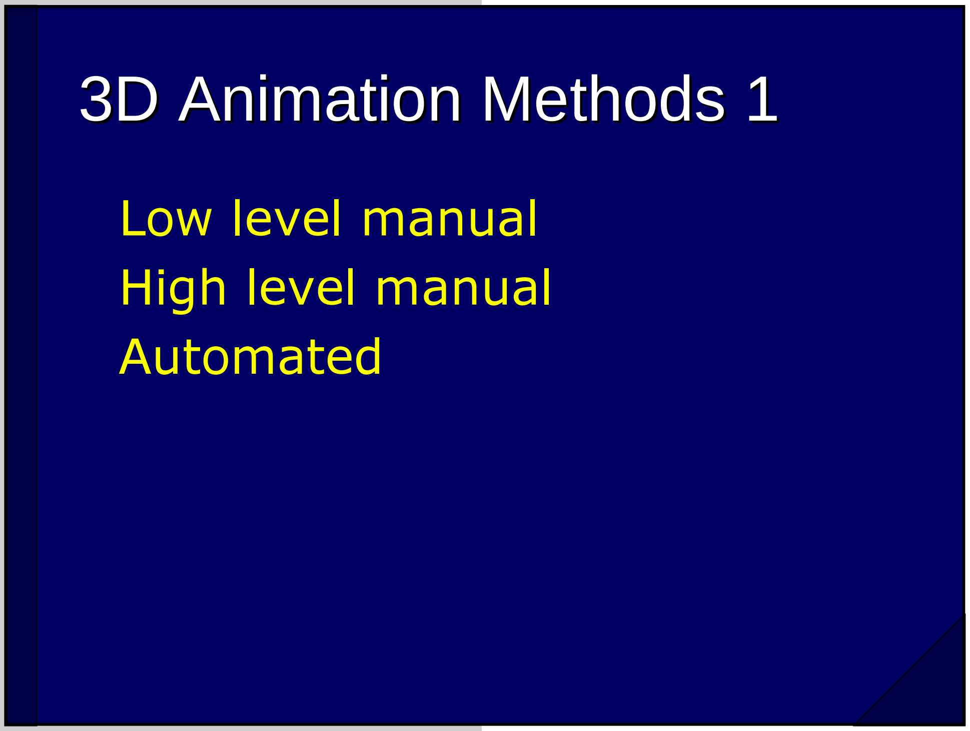
- FOV
- Focal length
- Position & orientation



Object Interpolation

- Parameterized transformation
  - Rotation
  - Scaling
  - Deformation
- Interpolate parameters
- Build new transformations



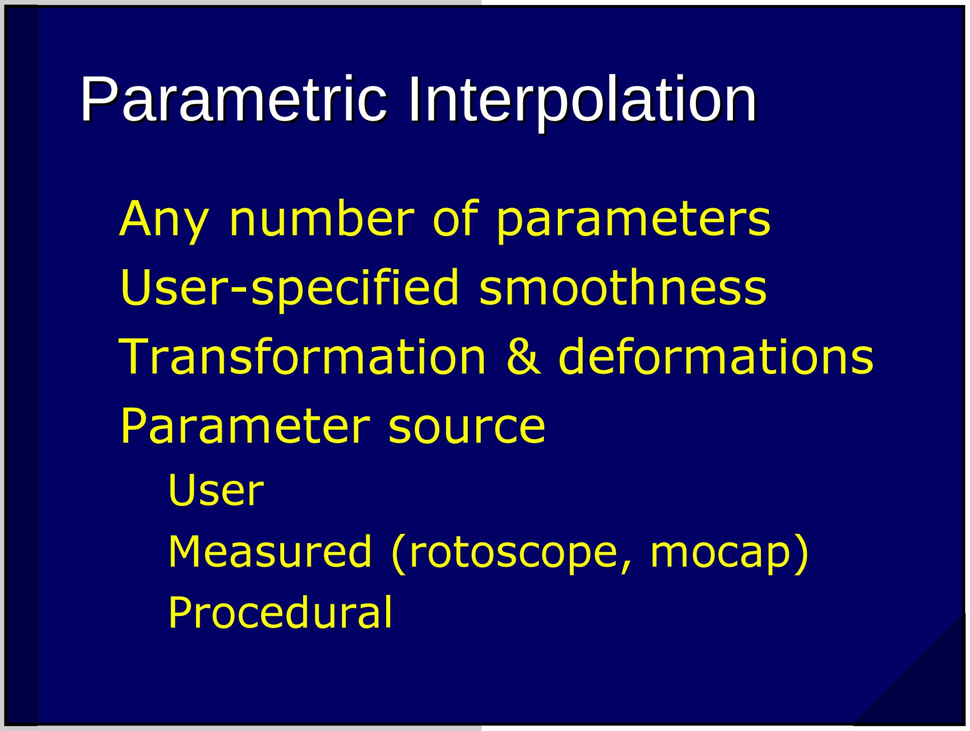


3D Animation Methods 1

Low level manual

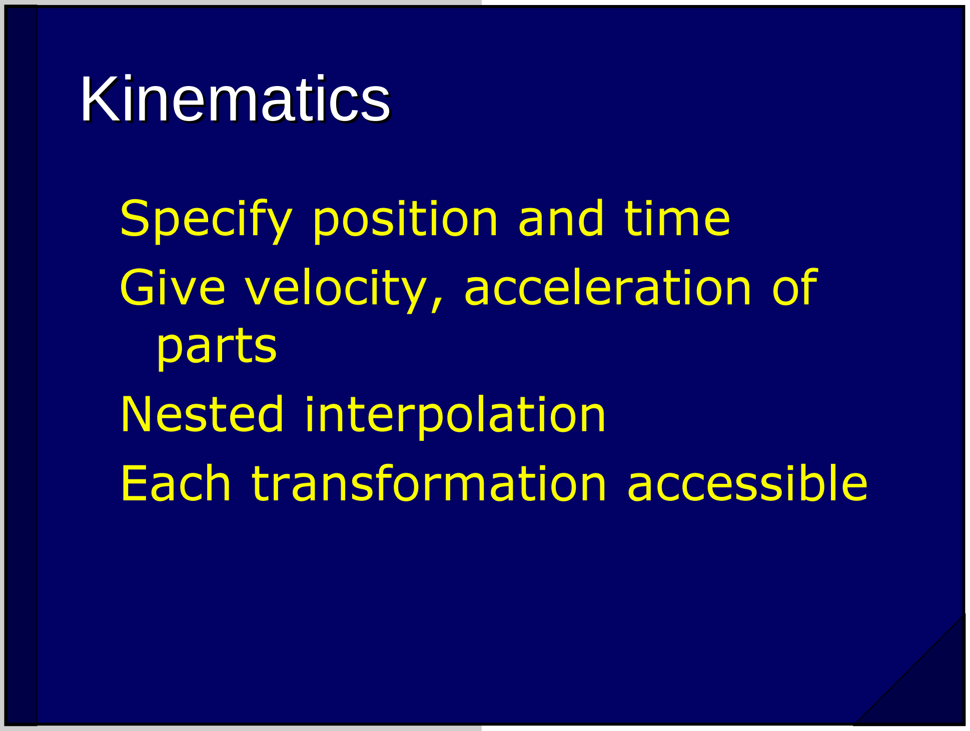
High level manual

Automated



Parametric Interpolation

- Any number of parameters
- User-specified smoothness
- Transformation & deformations
- Parameter source
  - User
  - Measured (rotoscope, mocap)
  - Procedural



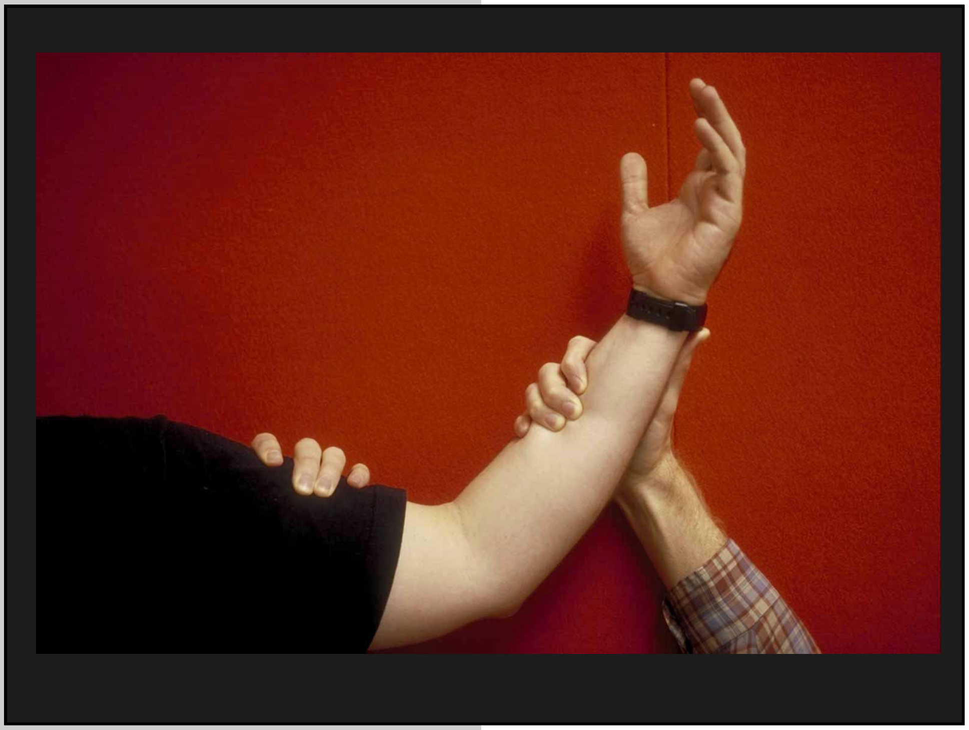
Kinematics

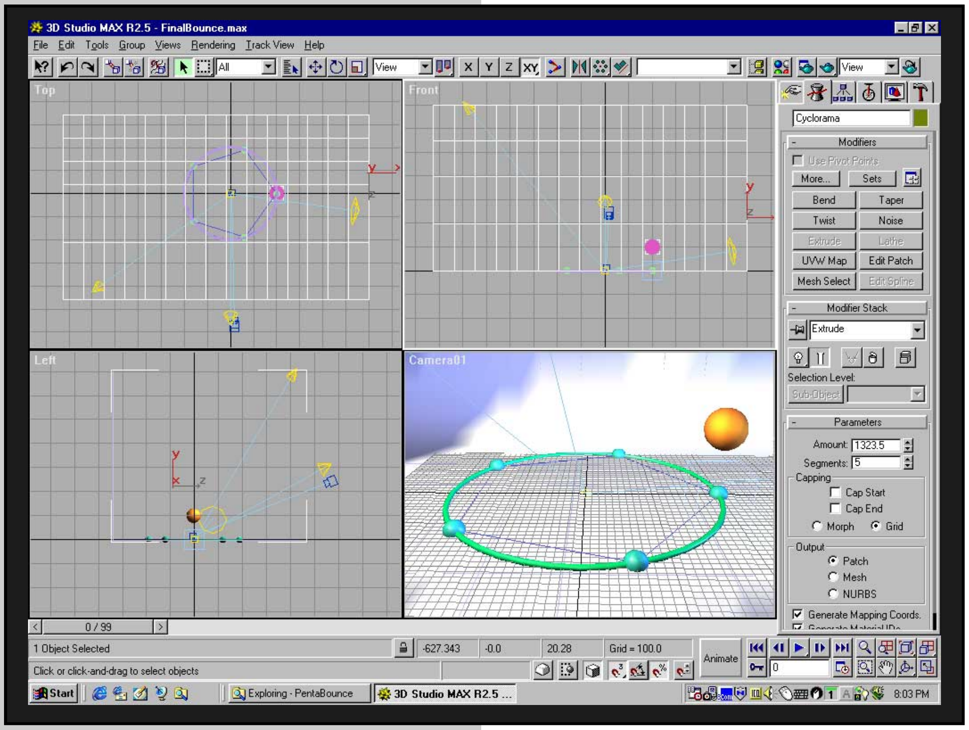
Specify position and time

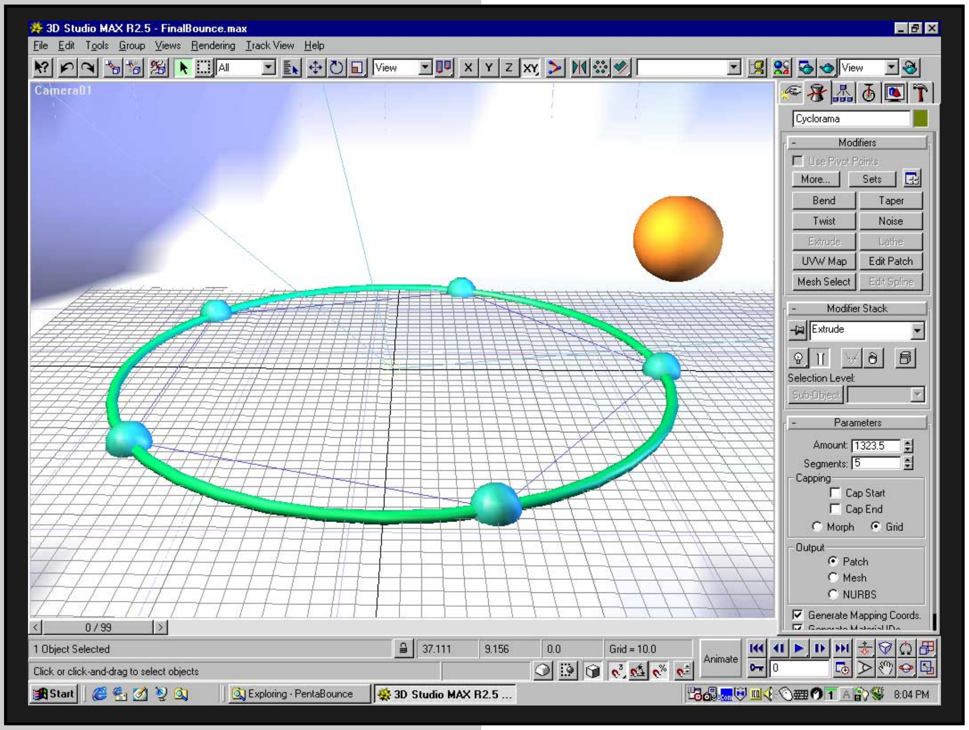
Give velocity, acceleration of parts

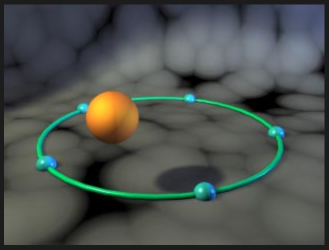
Nested interpolation

Each transformation accessible



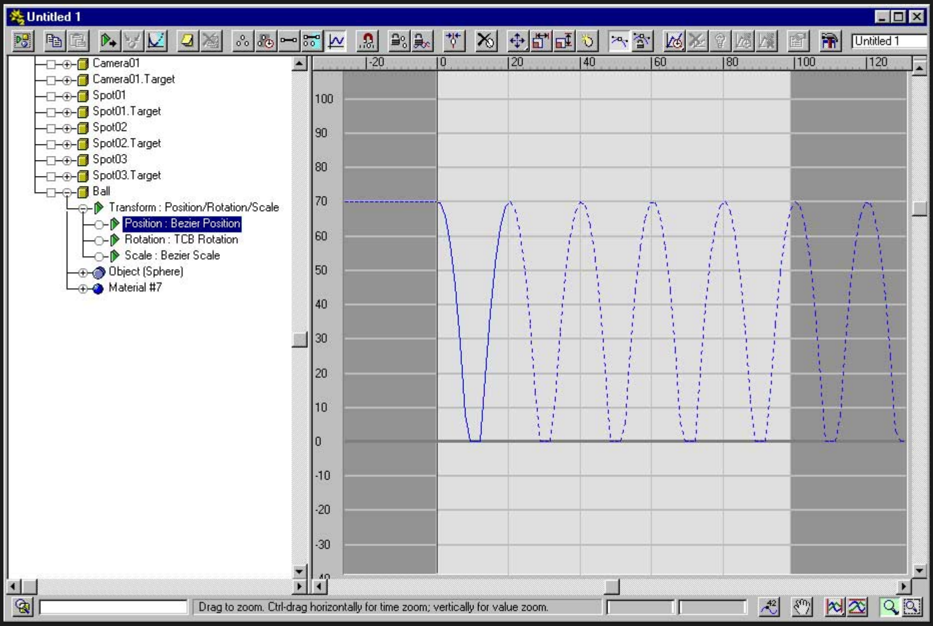

3D Studio MAX R2.5 - FinalBounce.max
File Edit Tools Group Views Rendering Track View Help
Top
Front
Left
Camera01
Cyclorama
Modifiers
More... Sets
Bend Taper
Twist Noise
Extrude Lathe
UVW Map Edit Patch
Mesh Select Edit Spline
Modifier Stack
Extrude
Selection Level:
Sub-Object
Parameters
Amount: 1323.5
Segments: 5
Capping
Cap Start
Cap End
Morph Grid
Output
Patch
Mesh
NURBS
Generate Mapping Coords.
0 / 99
1 Object Selected
Click or click-and-drag to select objects
-627.343 -0.0 20.28 Grid = 100.0
Animate
Start
Exploring - PentaBounce
3D Studio MAX R2.5 ...
8:03 PM


3D Studio MAX R2.5 - FinalBounce.max
File Edit Tools Group Views Rendering Track View Help
Camera01
Cyclorama
Modifiers
More... Sets
Bend Taper
Twist Noise
Extrude Lathe
UVW Map Edit Patch
Mesh Select Edit Spline
Modifier Stack
Extrude
Selection Level:
Sub-Object
Parameters
Amount: 1323.5
Segments: 5
Capping
Cap Start
Cap End
Morph Grid
Output
Patch
Mesh
NURBS
Generate Mapping Coords.
0 / 99
1 Object Selected
Click or click-and-drag to select objects
37.111 9.156 0.0 Grid = 10.0
Animate
Start
Exploring - PentaBounce
3D Studio MAX R2.5 ...
8:04 PM

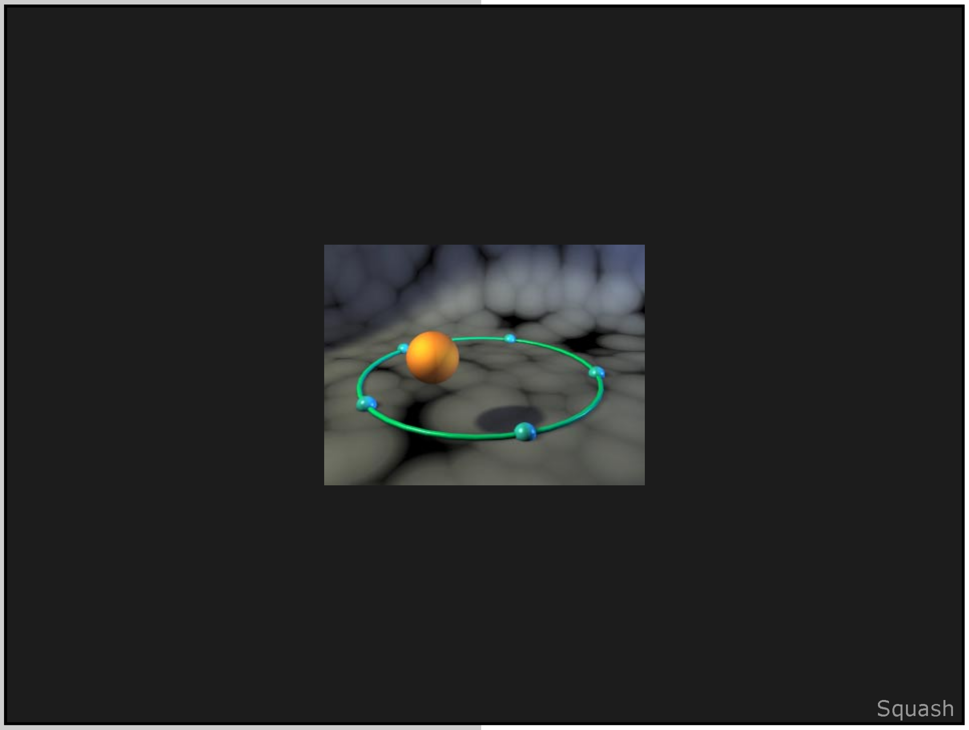


Bounce

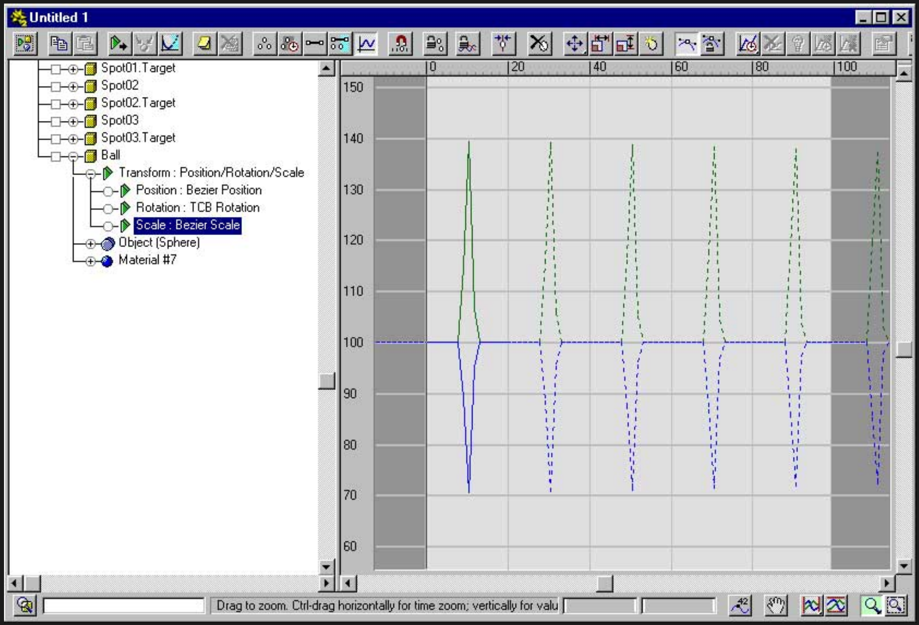
Ball vertical position

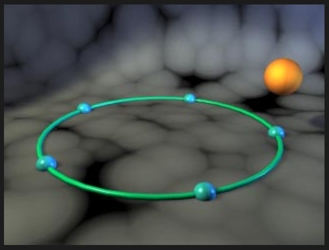


Drag to zoom. Ctrl-drag horizontally for time zoom; vertically for value zoom.



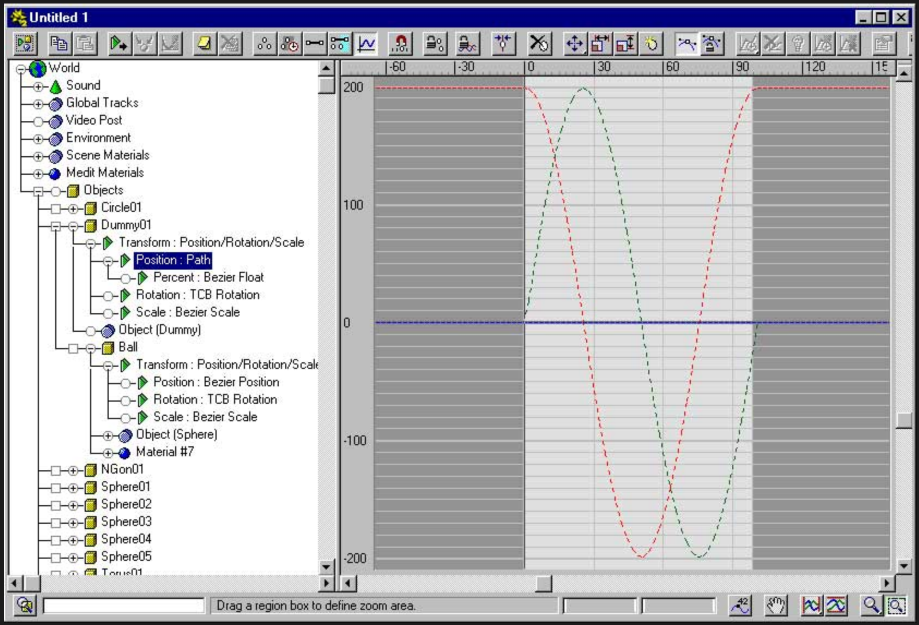
Ball squash percentage





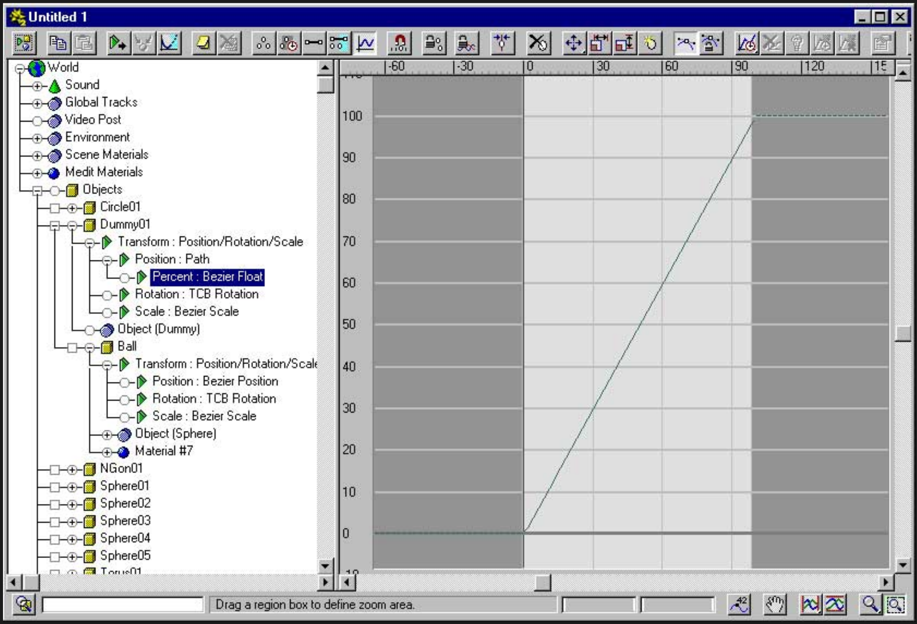
Path

Ball track position

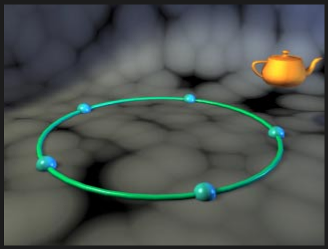


Drag a region box to define zoom area.

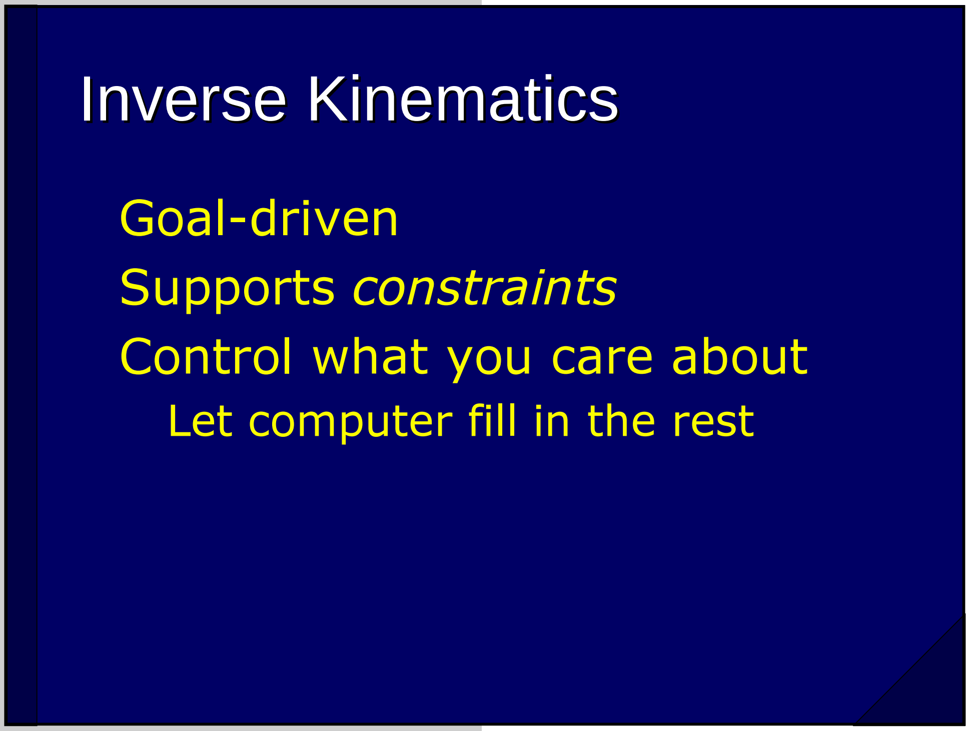
Ball track percent



Drag a region box to define zoom area.

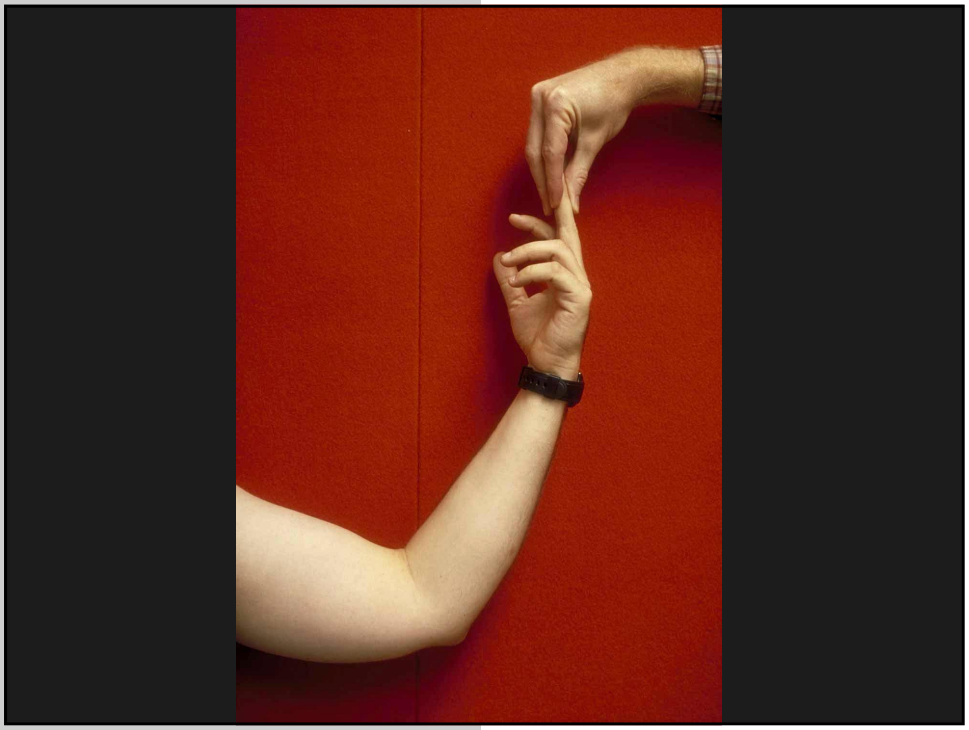


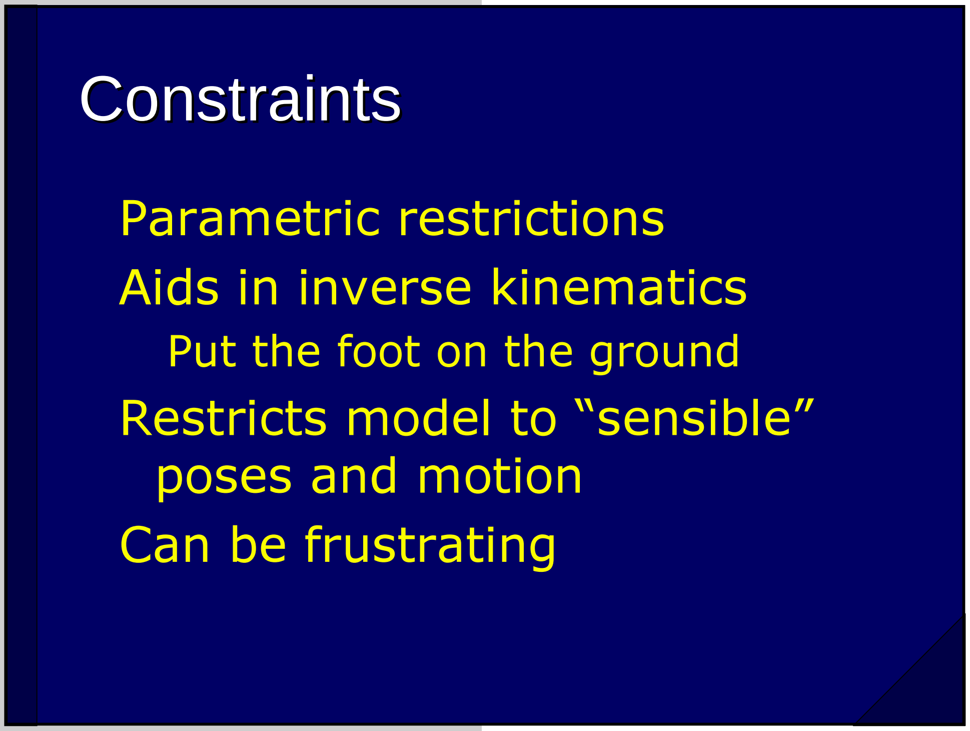
Path



Inverse Kinematics

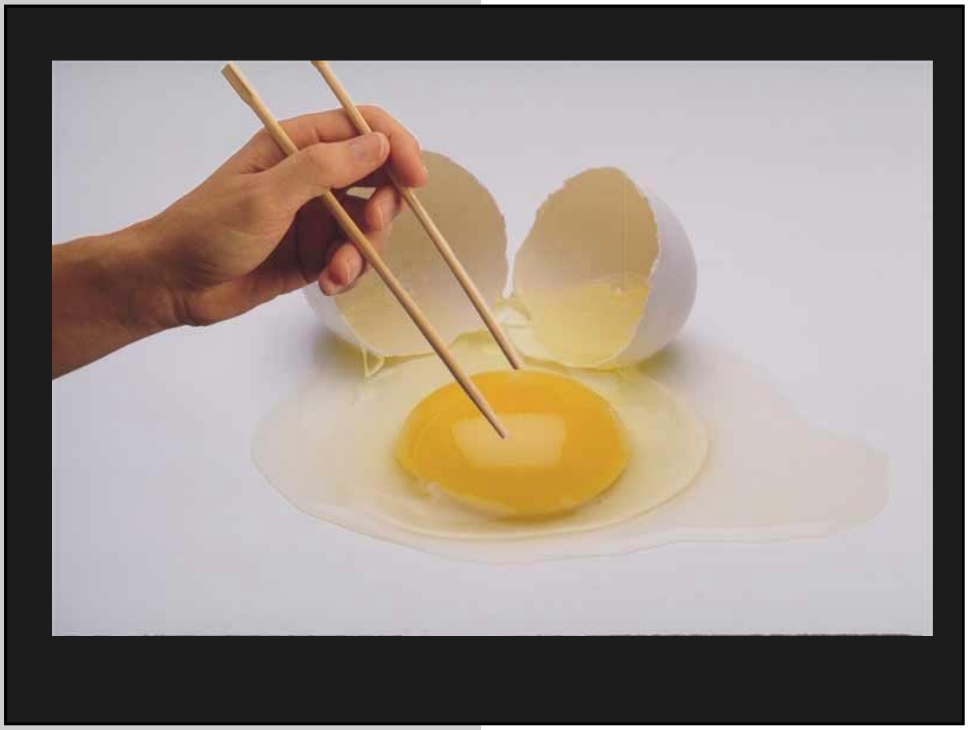
- Goal-driven
- Supports *constraints*
- Control what you care about
  - Let computer fill in the rest

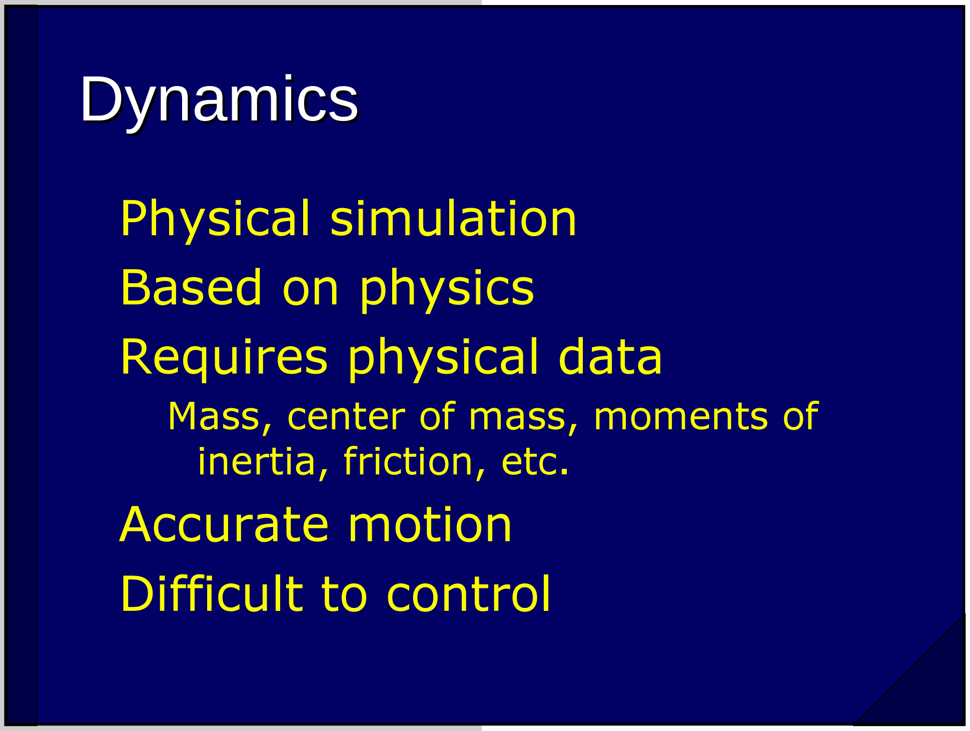




Constraints

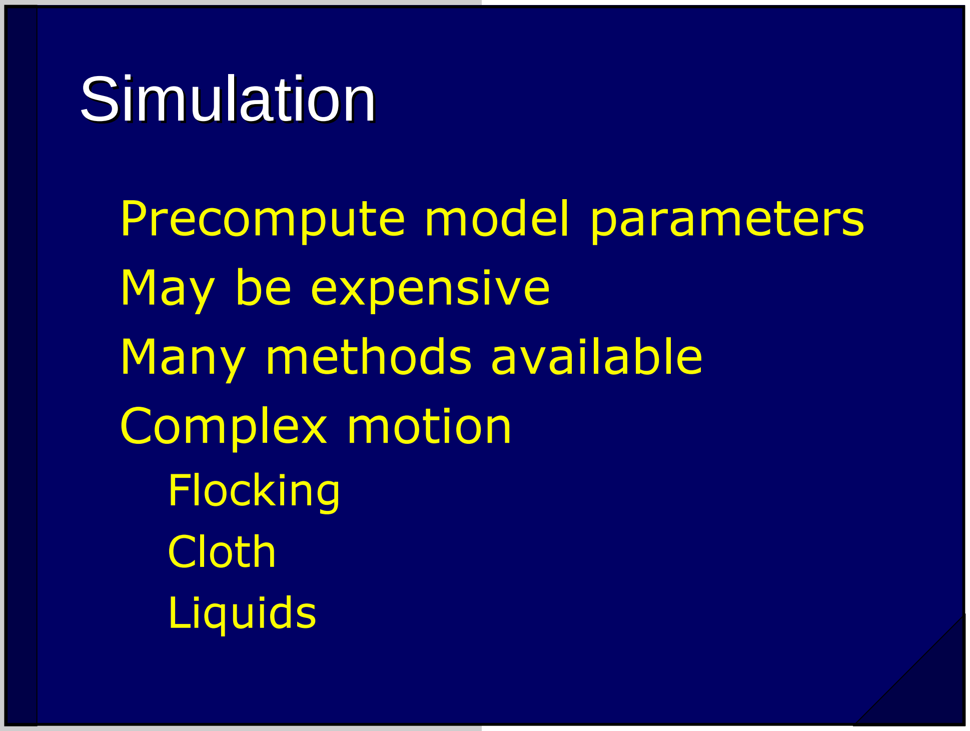
- Parametric restrictions
- Aids in inverse kinematics
  - Put the foot on the ground
- Restricts model to "sensible" poses and motion
- Can be frustrating





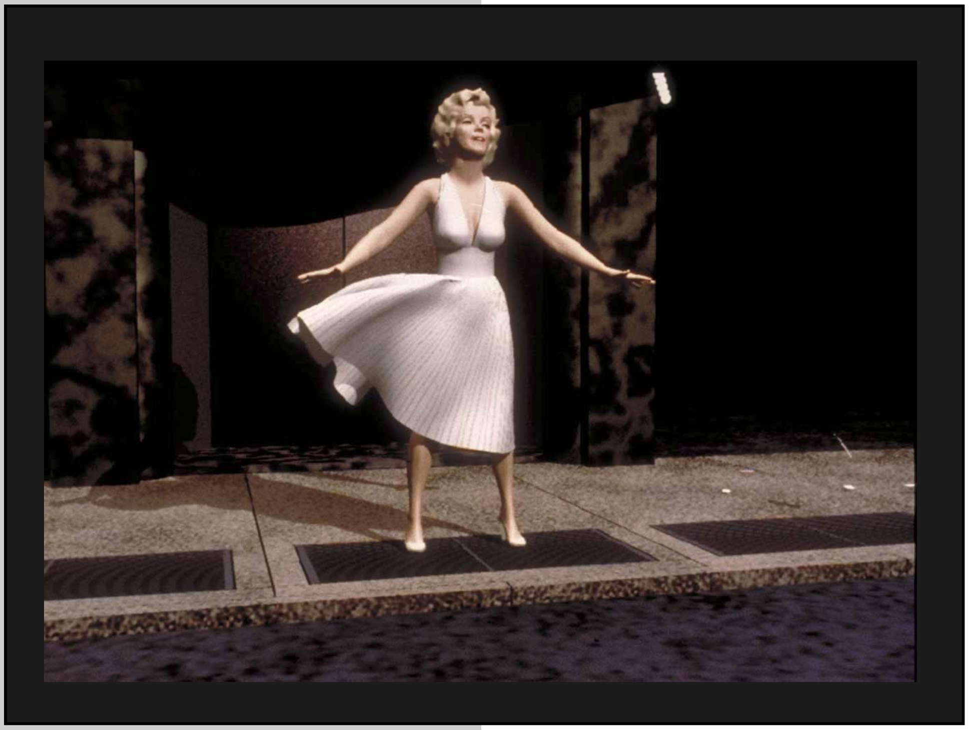
Dynamics

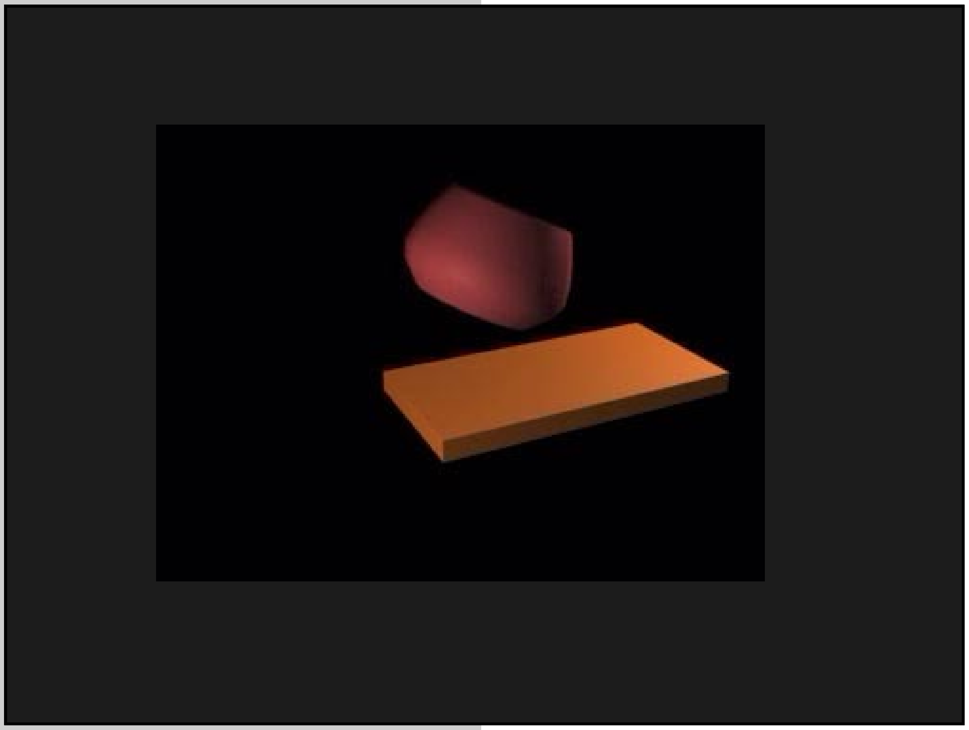
- Physical simulation
- Based on physics
- Requires physical data
  - Mass, center of mass, moments of inertia, friction, etc.
- Accurate motion
- Difficult to control

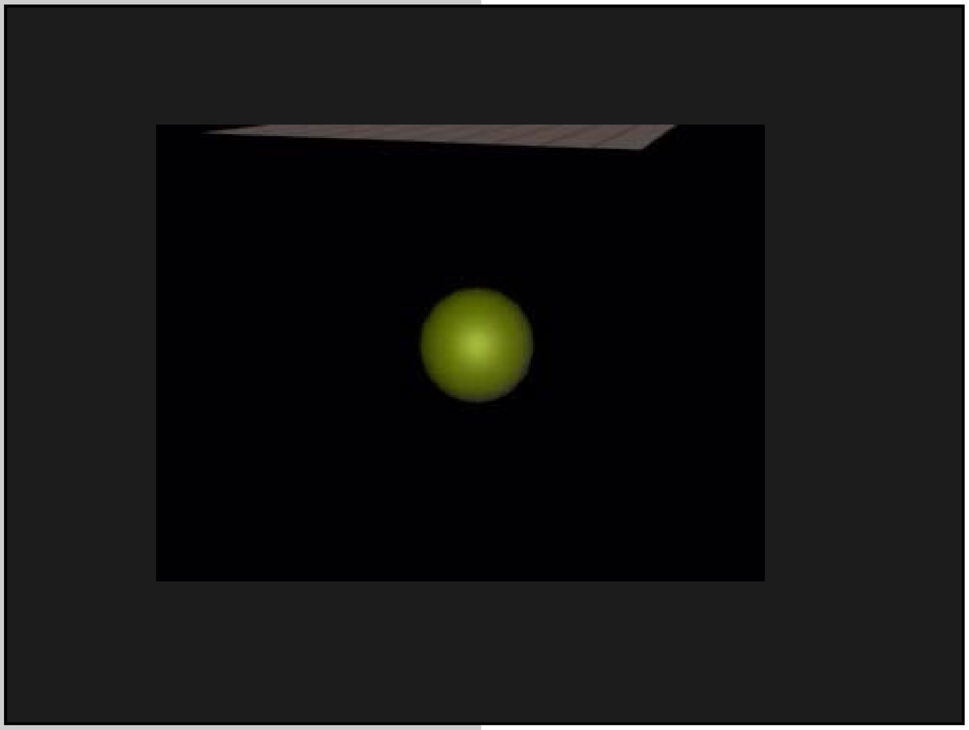


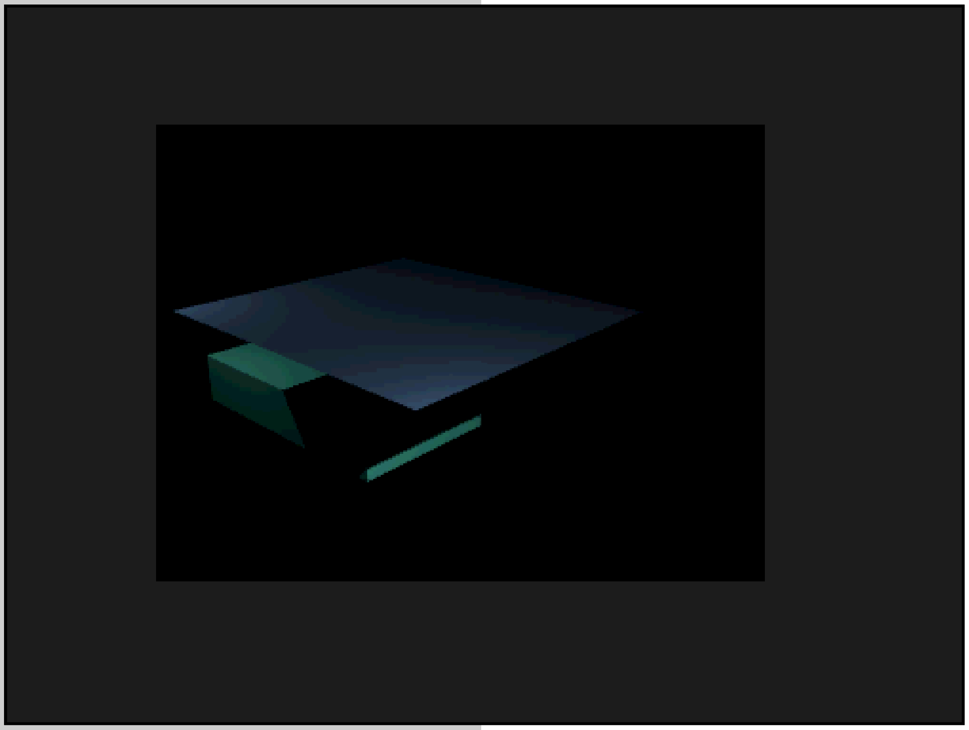
Simulation

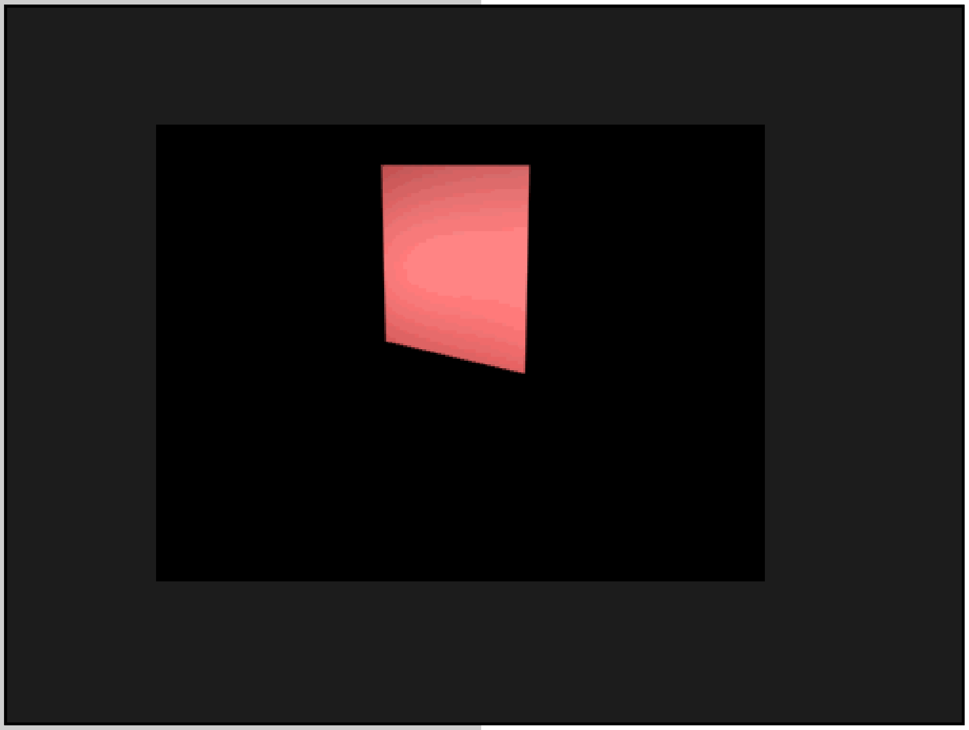
- Precompute model parameters
- May be expensive
- Many methods available
- Complex motion
  - Flocking
  - Cloth
  - Liquids

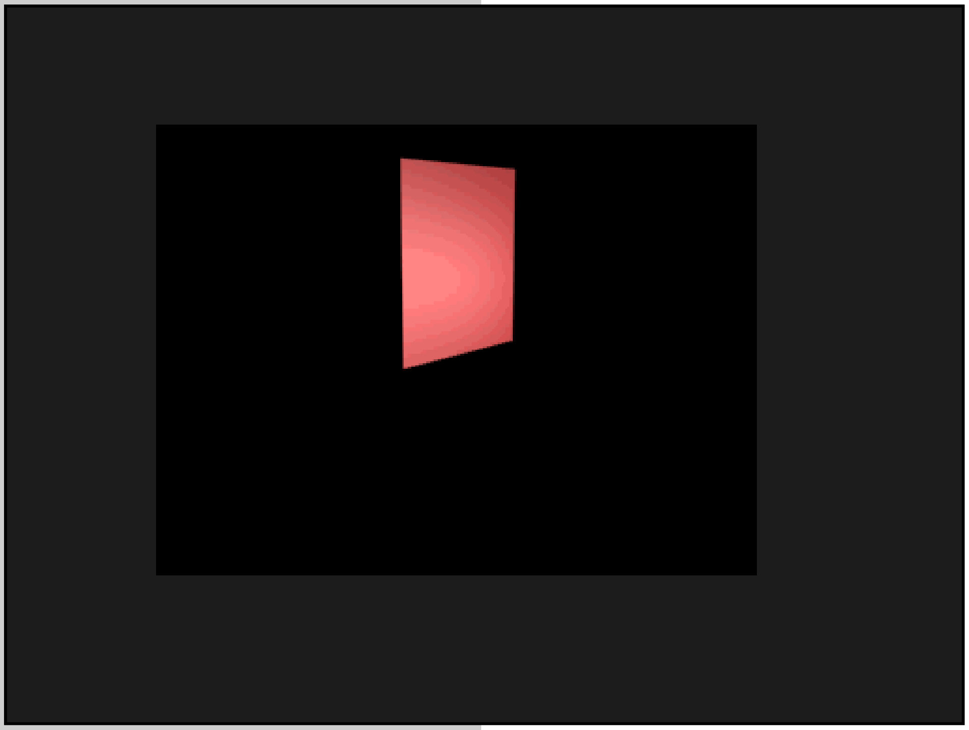


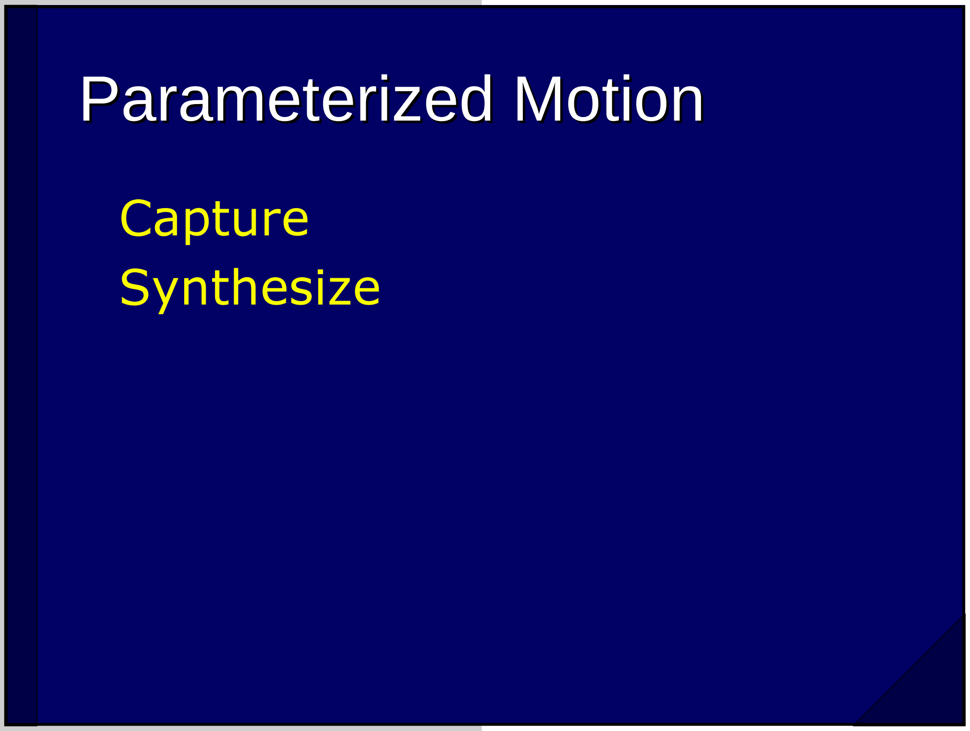






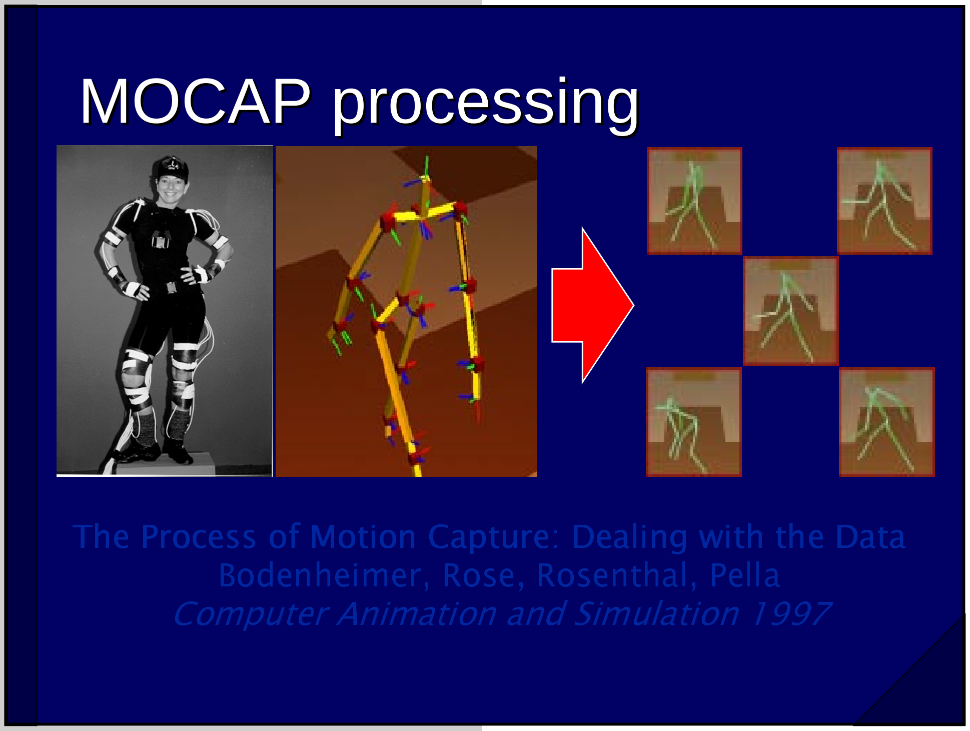






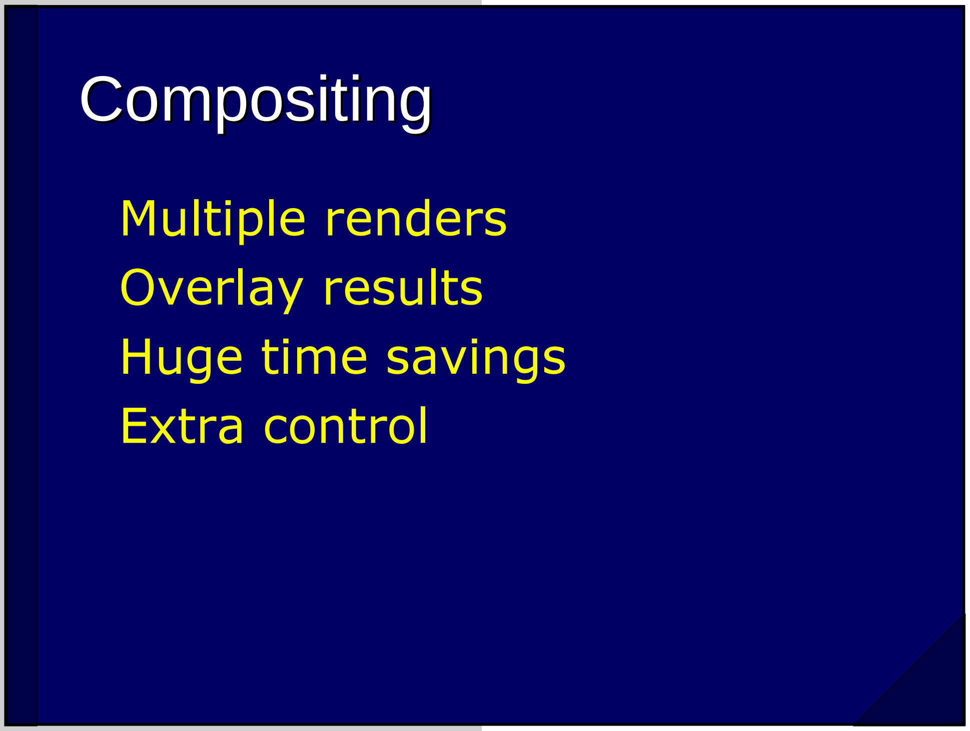
Parameterized Motion

- Capture
- Synthesize



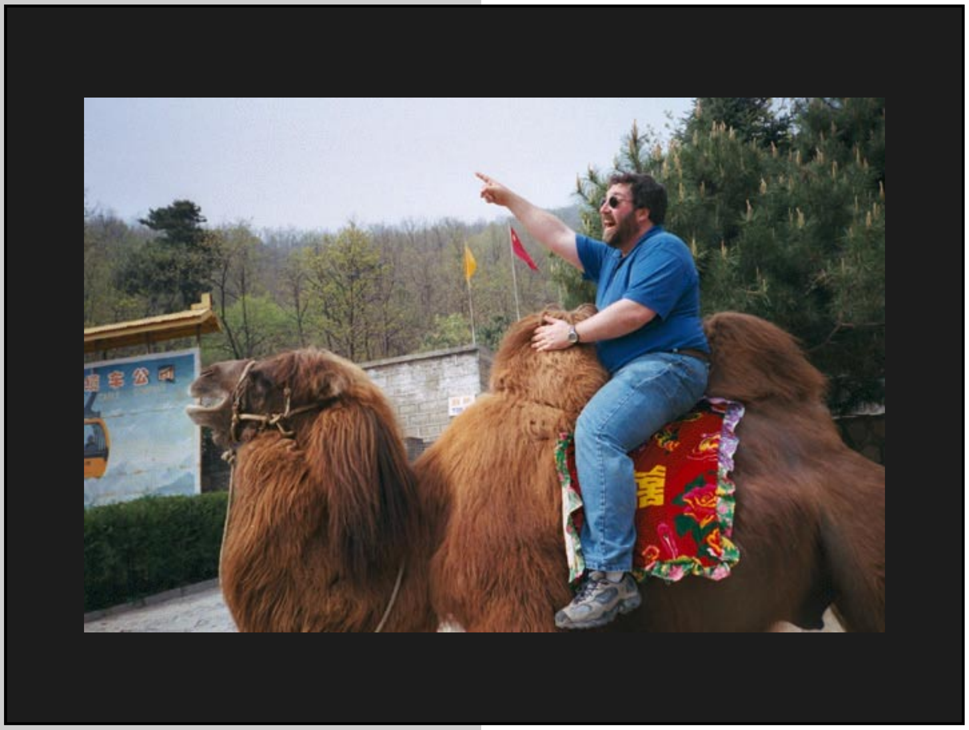
MOCAP processing

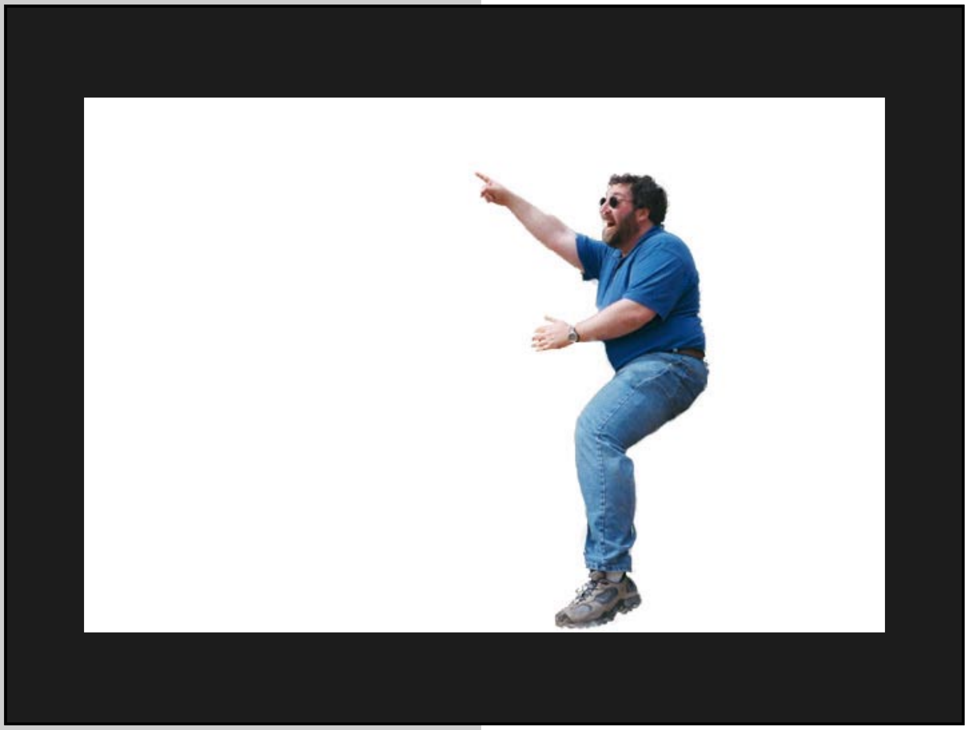
The Process of Motion Capture: Dealing with the Data
Bodenheimer, Rose, Rosenthal, Pella
Computer Animation and Simulation 1997

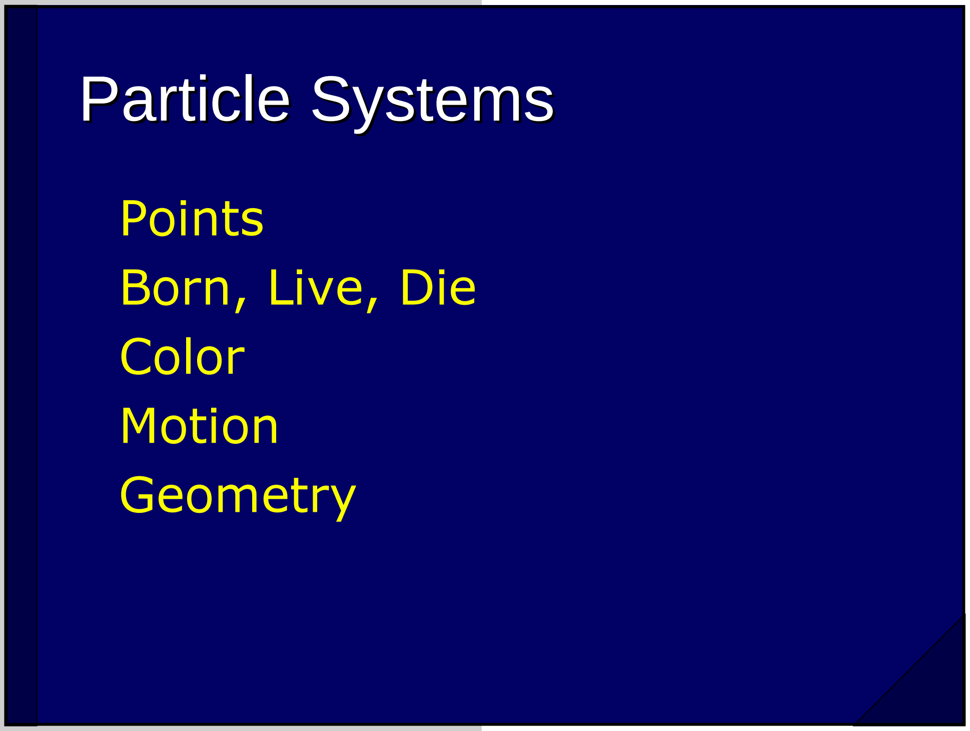


Compositing

Multiple renders
Overlay results
Huge time savings
Extra control

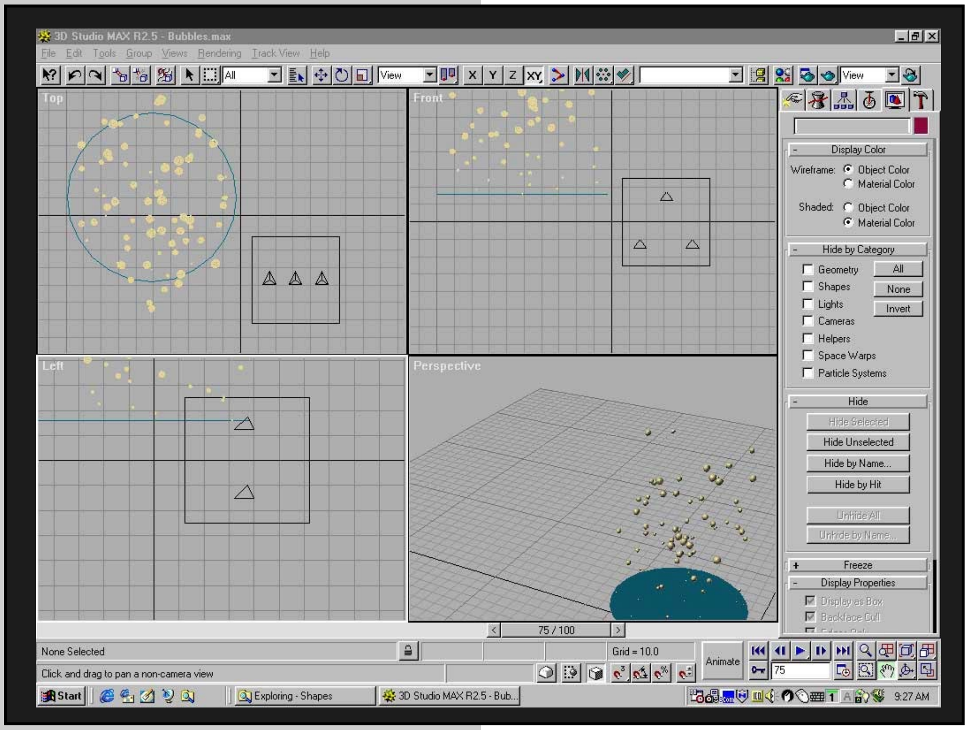


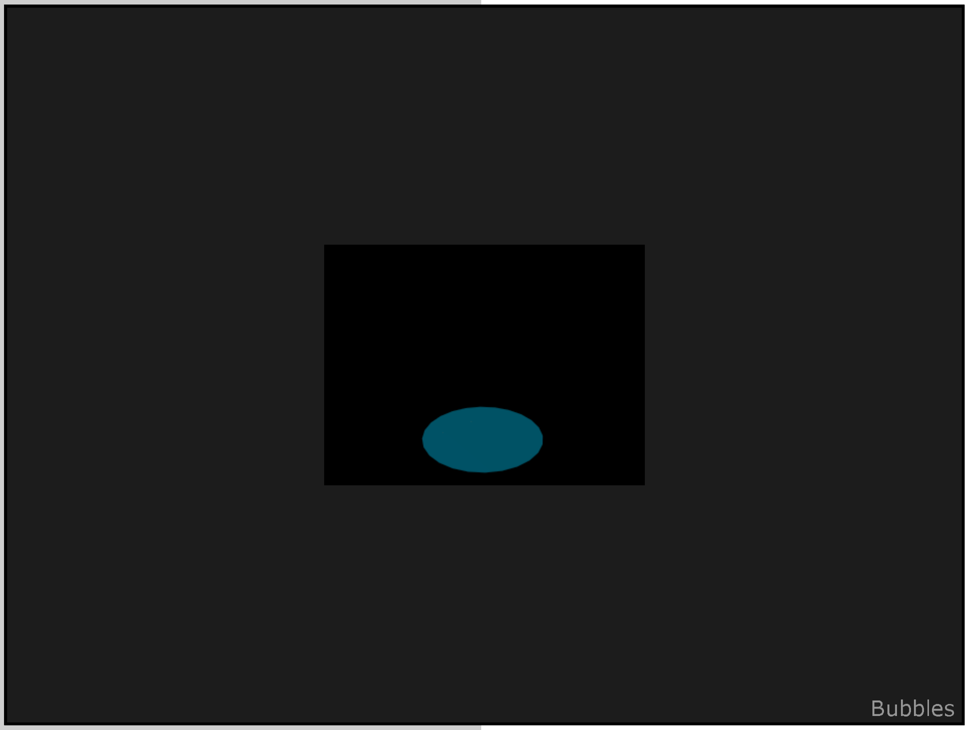


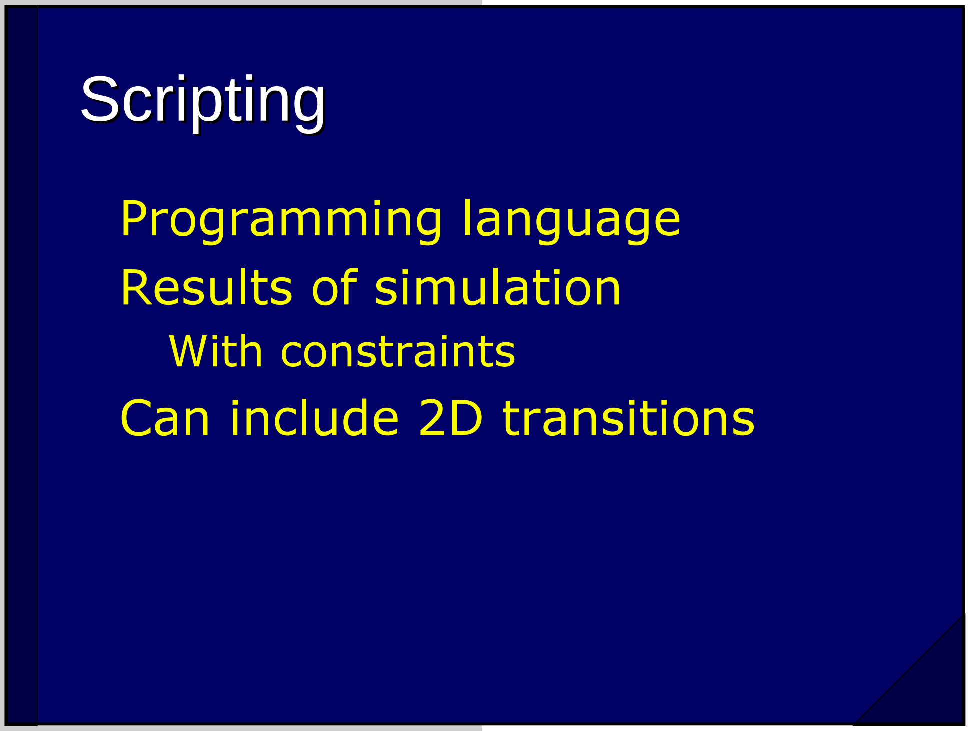


Particle Systems

- Points
- Born, Live, Die
- Color
- Motion
- Geometry

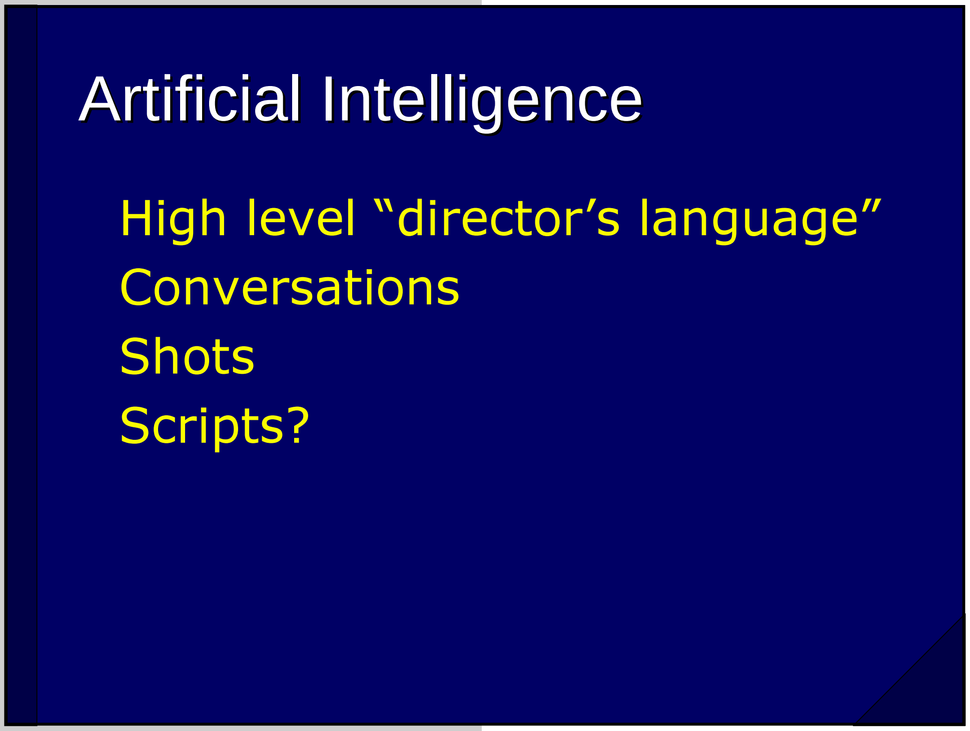

3D Studio MAX R2.5 - Bubbles.max
File Edit Tools Group Views Rendering Track View Help
Top
Front
Left
Perspective
Display Color
Wireframe: Object Color Material Color
Shaded: Object Color Material Color
Hide by Category
Geometry Shapes Lights Cameras Helpers Space Warps Particle Systems
All None Invert
Hide
Hide Selected Hide Unselected Hide by Name... Hide by Hit
Unhide All Unhide by Name...
Freeze
Display Properties
75 / 100
None Selected
Click and drag to pan a non-camera view
Grid = 10.0
Animate
Start
9:27 AM


Bubbles



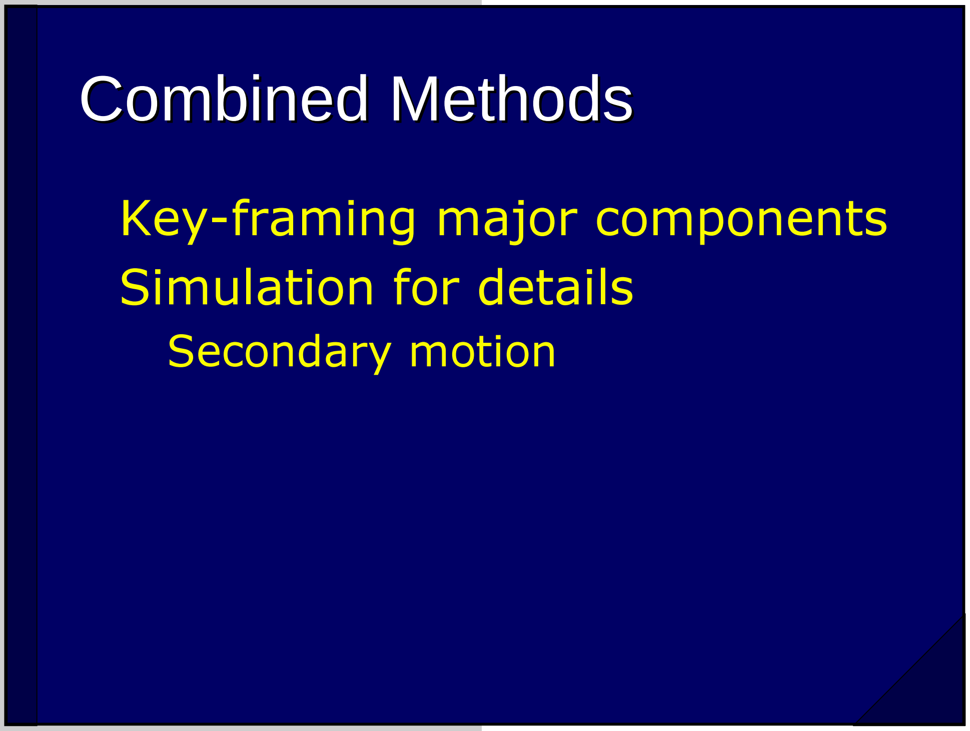
Scripting

- Programming language
- Results of simulation
  - With constraints
- Can include 2D transitions



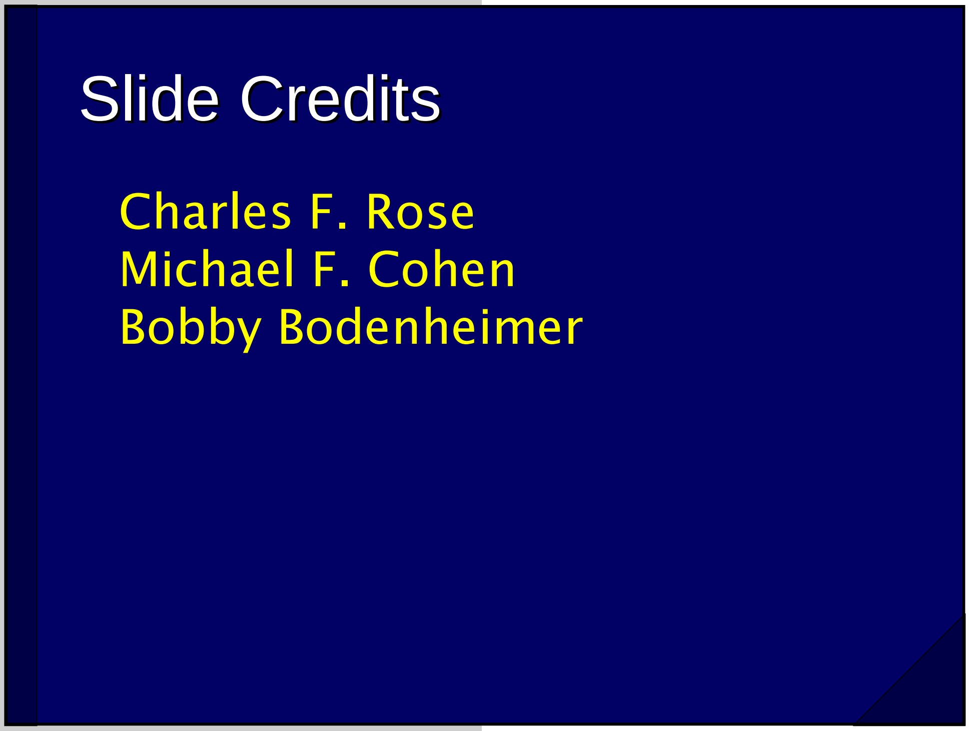
Artificial Intelligence

- High level “director’s language”
- Conversations
- Shots
- Scripts?



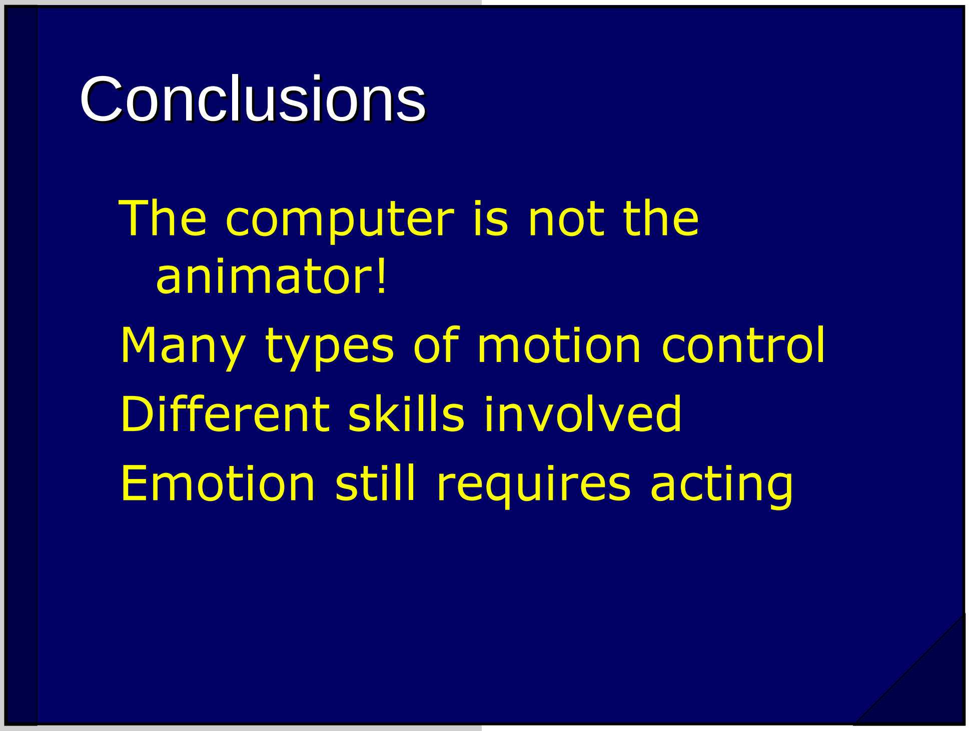
Combined Methods

- Key-framing major components
- Simulation for details
  - Secondary motion



Slide Credits

Charles F. Rose
Michael F. Cohen
Bobby Bodenheimer



Conclusions

The computer is not the animator!

Many types of motion control

Different skills involved

Emotion still requires acting

Computer Graphics for Scientific Visualization

Mike Bailey

San Diego Supercomputer Center
University of California San Diego

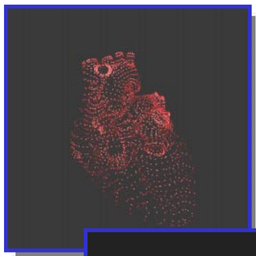
`mjb@sdsc.edu`

`http://www.sdsc.edu/~mjb`

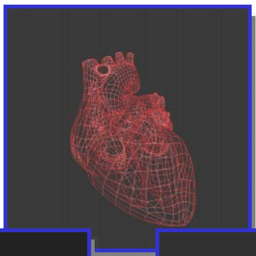
University of California, San Diego
San Diego Supercomputer Center

A Gallery of Display Options

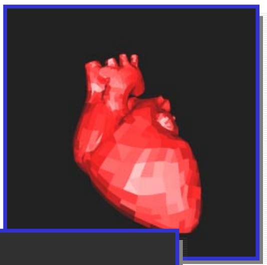
Points



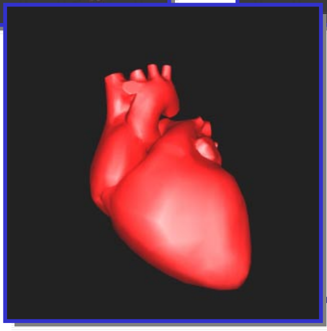
Lines



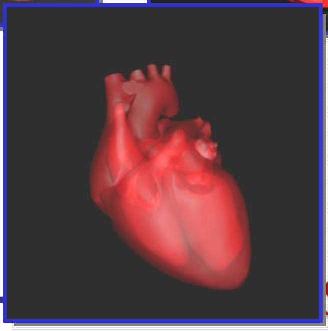
Flat Shaded

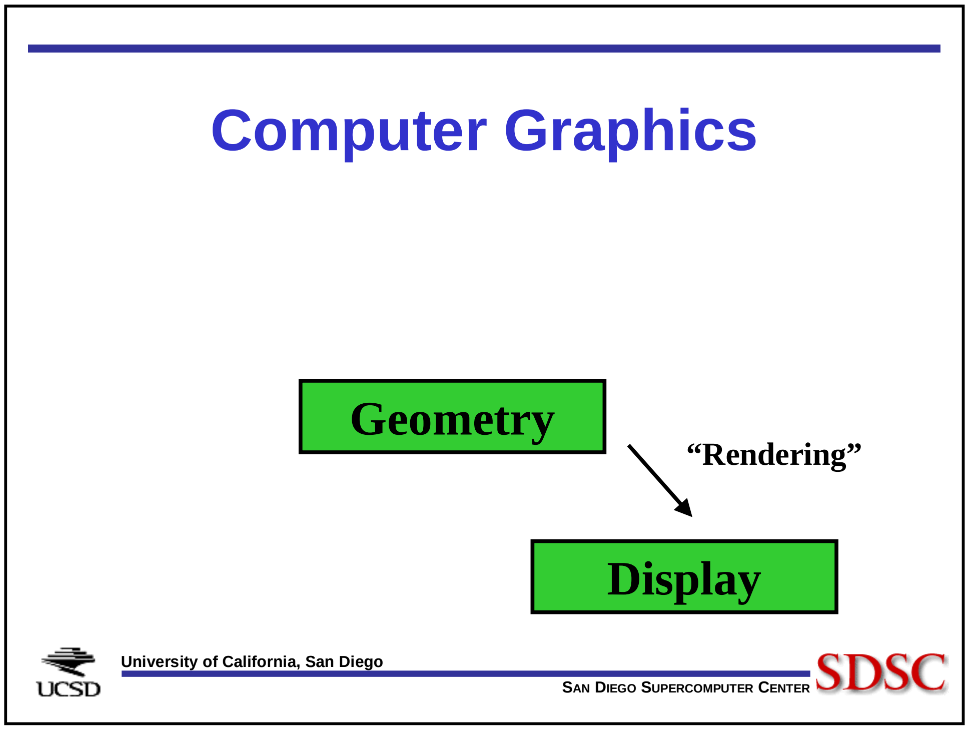


Smooth Shaded



Transparent





Computer Graphics

Geometry

“Rendering”

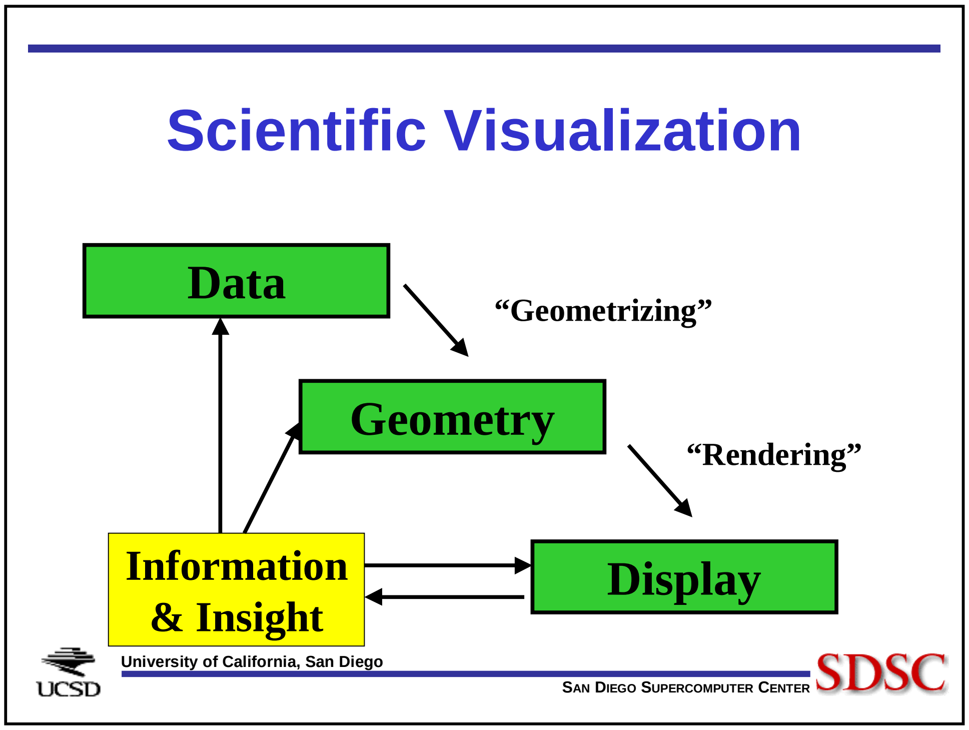
Display

University of California, San Diego

UCSD

SAN DIEGO SUPERCOMPUTER CENTER

SDSC



Scientific Visualization

Data

“Geometrizing”

Geometry

“Rendering”

Information & Insight

Display

University of California, San Diego

UCSD

SAN DIEGO SUPERCOMPUTER CENTER

SDSC

Visualization Dimensions

Spatial Dimension: **How many numbers required to locate a data point in space**

Data Dimension: **How many data numbers are at each data point (also called *rank*)**

1D = *scalar*

2D or 3D = *vector*

Visualization Dimensions

Data Dimension

Spatial Dimension		Scalar	Vector
	2D	2D data values	2D flow
	3D	3D data values	3D flow

1D Data Dimension
2D Spatial Dimension

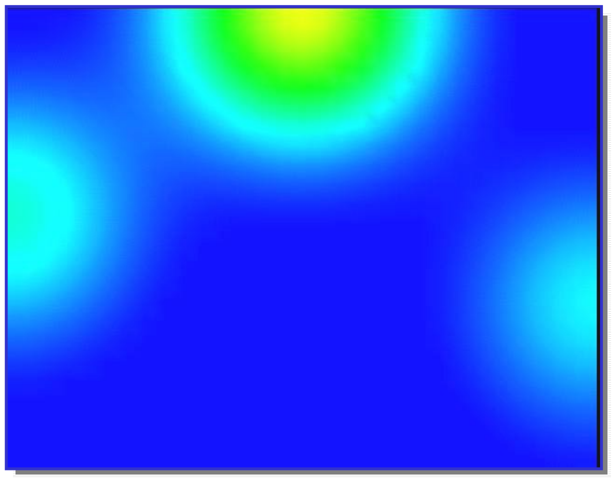
Examples:

- Temperatures on a plane
- Barometric pressures on a map
- Heights on a map

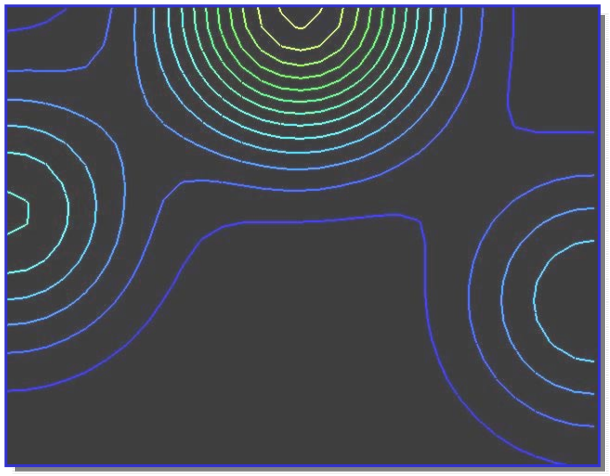
Techniques:

- Colored dots
- Contour lines
- Interpolated Colors

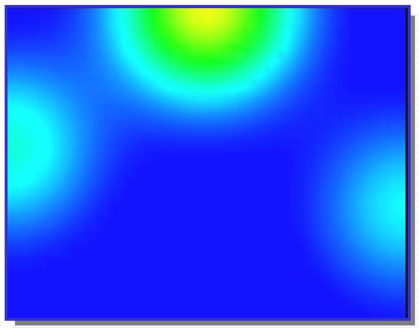
2D Color Interpolation

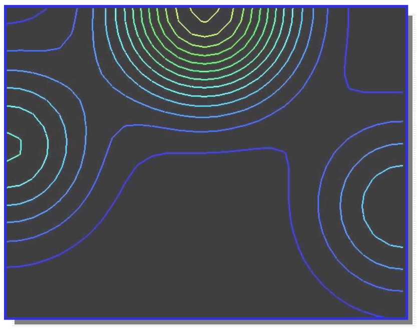


2D Contours



Same Data, Different Geometrization





1D Data Dimension
3D Spatial Dimension

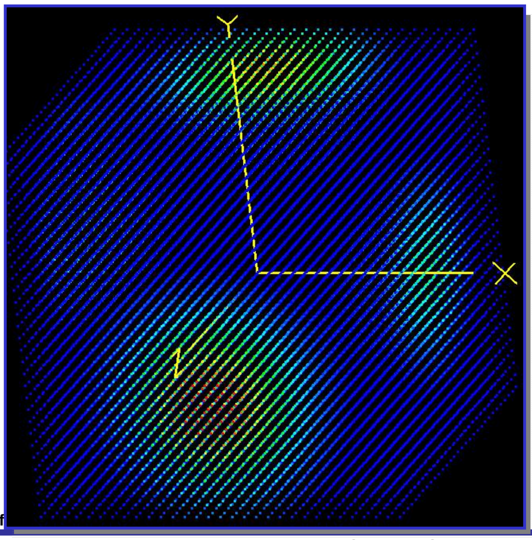
Examples:

- Temperatures in a room
- Molecular potentials

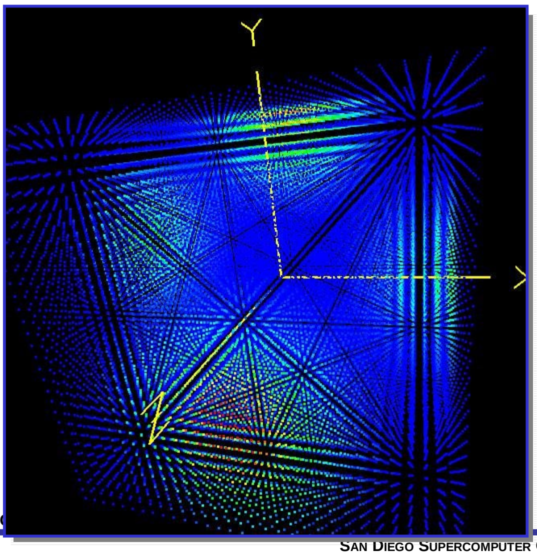
Techniques:

- Colored dots ("point cloud")
- Cutting plane
- Isosurfaces

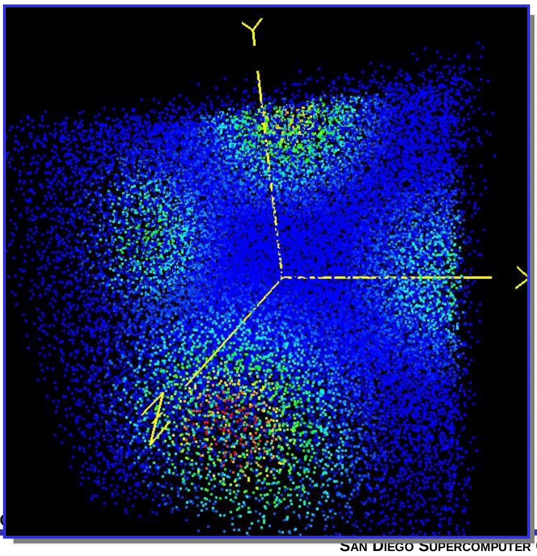
3D Orthographic Point Cloud



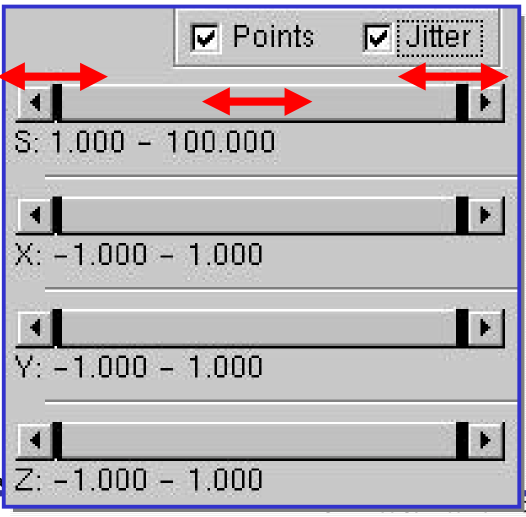
3D Perspective Point Cloud



3D Jittered Point Cloud



Cropping the Point Cloud: Range Sliders



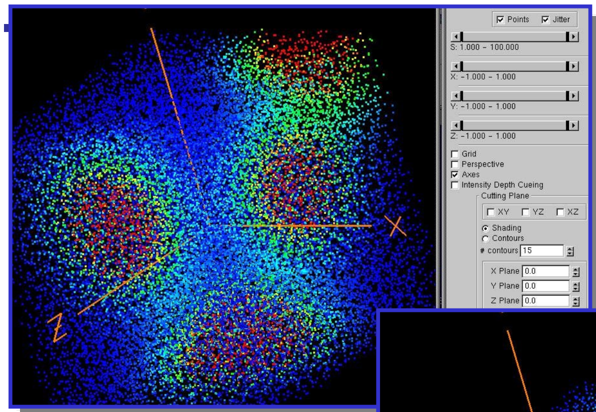
☑ Points ☑ Jitter

S: 1.000 – 100.000

X: −1.000 – 1.000

Y: −1.000 – 1.000

Z: −1.000 – 1.000



☑ Points ☑ Jitter

S: 1.000 – 100.000

X: −1.000 – 1.000

Y: −1.000 – 1.000

Z: −1.000 – 1.000

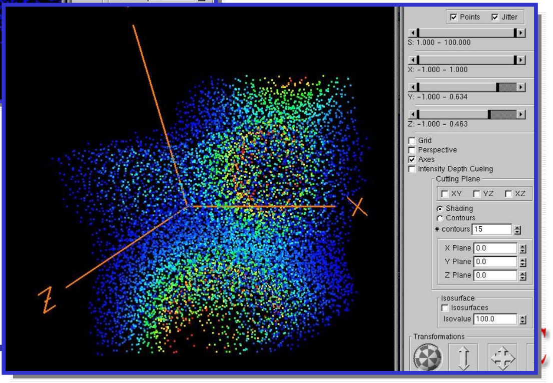
☐ Grid
☐ Perspective
☑ Axes
☐ Intensity Depth Cueing

Cutting Plane
☐ XY ☐ YZ ☐ XZ

◉ Shading
○ Contours
# contours 15

X Plane 0.0
Y Plane 0.0
Z Plane 0.0

Cropping in X, Y, Z



☑ Points ☑ Jitter

S: 1.000 – 100.000

X: −1.000 – 1.000

Y: −1.000 – 0.634

Z: −1.000 – 0.463

☐ Grid
☐ Perspective
☑ Axes
☐ Intensity Depth Cueing

Cutting Plane
☐ XY ☐ YZ ☐ XZ

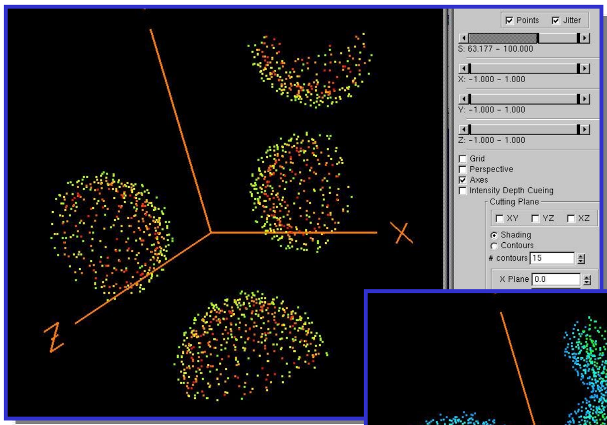
◉ Shading
○ Contours
# contours 15

X Plane 0.0
Y Plane 0.0
Z Plane 0.0

Isosurface
☐ Isosurfaces
Isovalue 100.0

Transformations

Cropping in S



☑ Points ☑ Jitter

S: 63.177 – 100.000

X: –1.000 – 1.000

Y: –1.000 – 1.000

Z: –1.000 – 1.000

☐ Grid
☐ Perspective
☑ Axes
☐ Intensity Depth Cueing

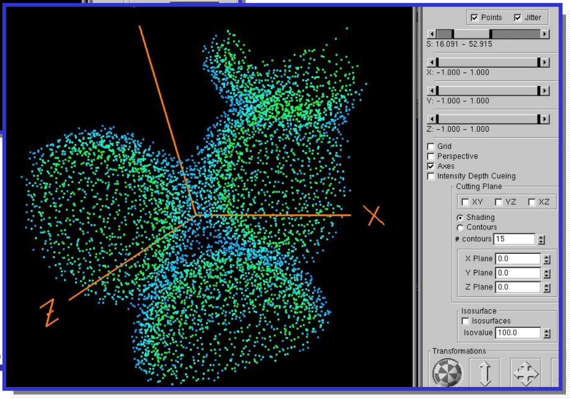
Cutting Plane

☐ XY ☐ YZ ☐ XZ

[✓ Shading] Contours

# contours 15

X Plane 0.0



☑ Points ☑ Jitter

S: 16.091 – 52.915

X: –1.000 – 1.000

Y: –1.000 – 1.000

Z: –1.000 – 1.000

☐ Grid
☐ Perspective
☑ Axes
☐ Intensity Depth Cueing

Cutting Plane

☐ XY ☐ YZ ☐ XZ

[✓ Shading] Contours

# contours 15

X Plane 0.0
Y Plane 0.0
Z Plane 0.0

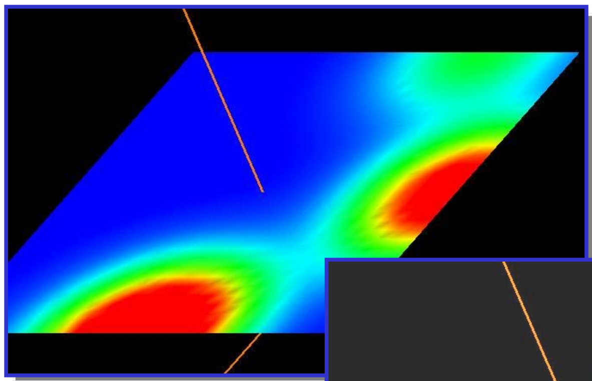
Isosurface

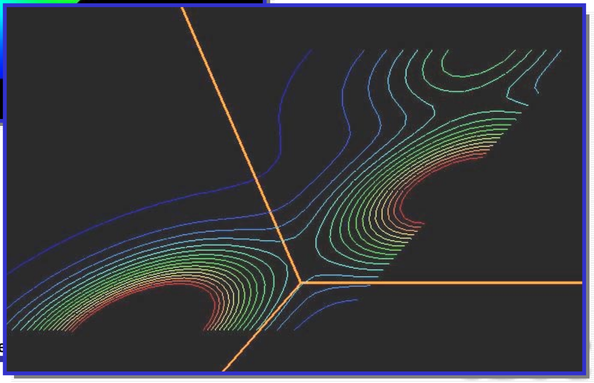
☐ Isosurfaces

Isovalue 100.0

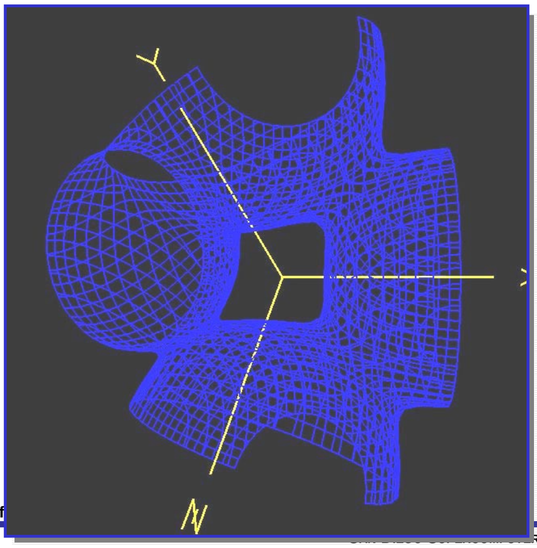
Transformations

What's Happening on a Cutting Plane?

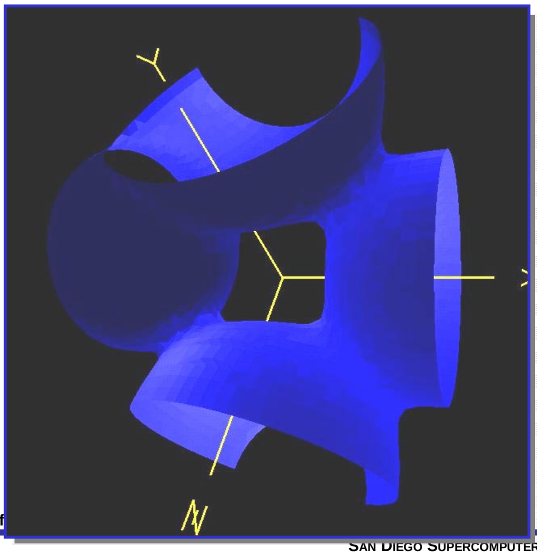




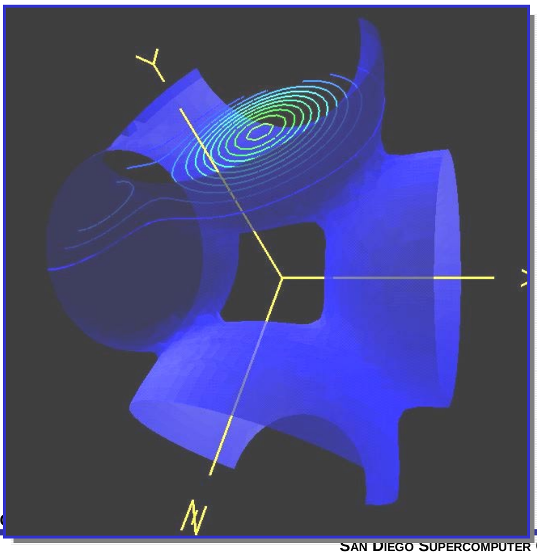
What's Happening within the Volume?
3D Wireframe Isosurface



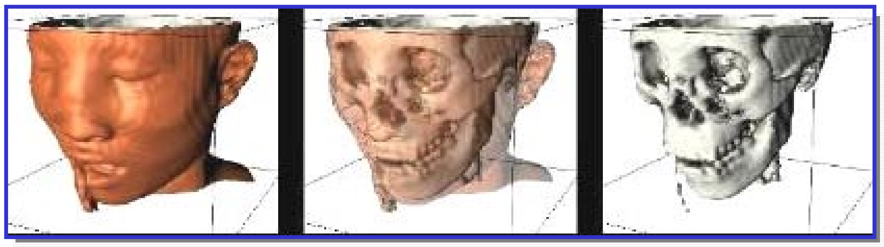
3D Polygonal Isosurface



Isosurface + Contour Plane



Isosurfaces of Medical Data



3D Data Dimension
3D Spatial Dimension

Examples:

- **3D flow field**

Techniques:

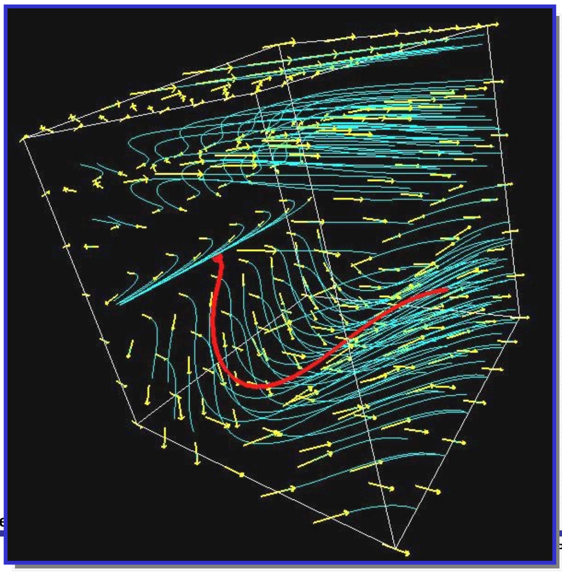
- **Arrows ("vector cloud")**
- **Streamlines**

University of California, San Diego

San Diego Supercomputer Center

SDSC

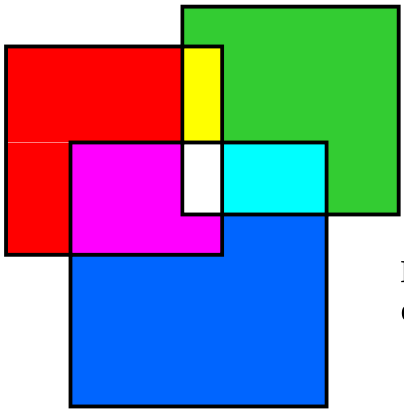
3D Flow Field



UCSD Unive

PUTER CENTER SDSC

Setting Colors



Red+Green=Yellow
Green+Blue=Cyan
Red+Blue=Magenta
Red+Green+Blue=White

Known as “Additive Color”, or “RGB”

```
glColor3f( r, g, b );
```

Color Rules: Your Eye

Rods: Provide perception of light intensity, or luminance. Lots of these, mostly concentrated in the periphery of your field of vision.

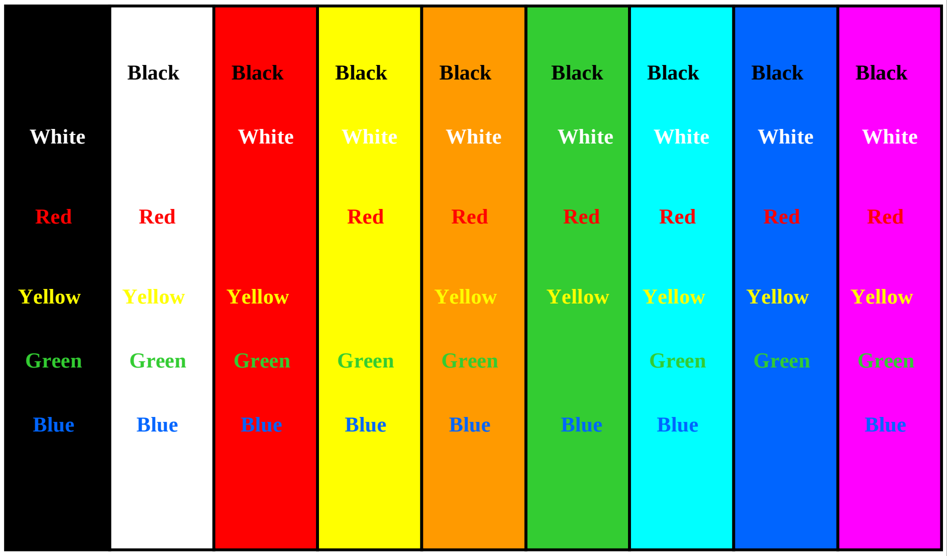
Cones: Provide perception of colors. Much fewer of these, mostly concentrated in the center of your field of vision.

Color Rules

- Remember the Luminance Equation:

$$L = .30R + .59G + .11B$$

Use it to decide what makes a good contrast and what doesn’t



	Black	Black	Black	Black	Black	Black	Black	Black
White		White	White	White	White	White	White	White
Red	Red		Red	Red	Red	Red	Red	Red
Yellow	Yellow	Yellow		Yellow	Yellow	Yellow	Yellow	Yellow
Green	Green	Green	Green	Green		Green	Green	Green
Blue	Blue	Blue	Blue	Blue	Blue	Blue		Blue

Color Rules

- Do not use Reds and Blues directly together

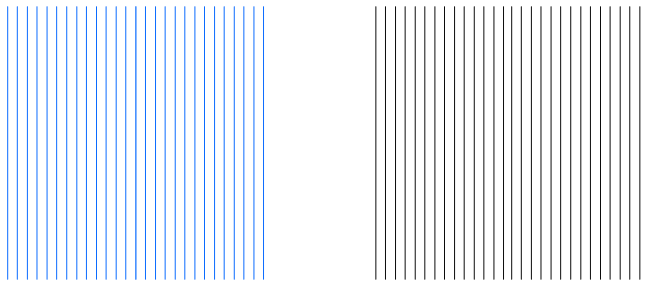
University of California, San Diego
San Diego Supercomputer Center

Reds and Blues are on opposite ends of the color spectrum. It is hard for your eyes to focus on both.

University of California, San Diego
San Diego Supercomputer Center

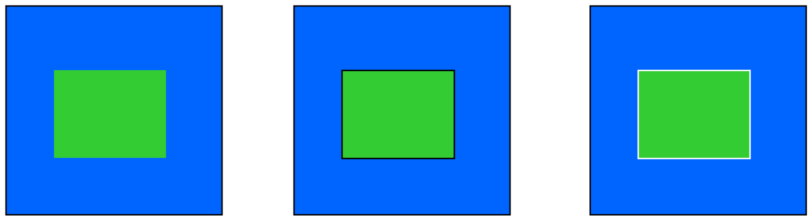
Color Rules

- The best colors for fine detail are black-grey-white. This is especially true if you are going to be videotaping this display.



Color Rules

- Surround contrasting color areas with a white or black line



University of California, San Diego

San Diego Supercomputer Center

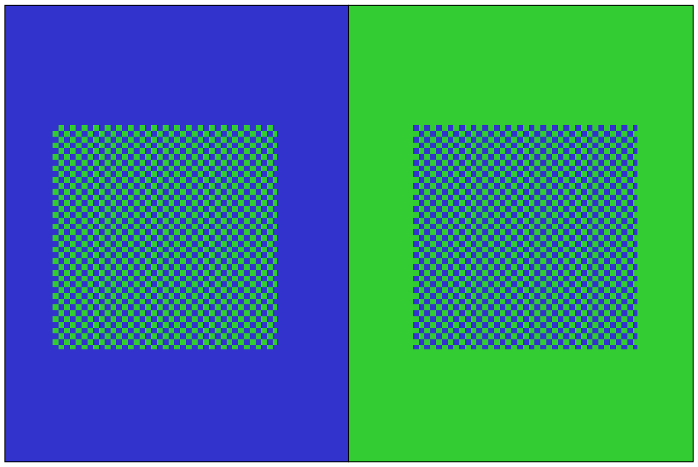
Color Rules

Be aware that our perception of color changes with:

- **The surrounding color**
- **The size of the object**
- **How close two objects are**
- **The ambient light**
- **The age of the viewer**

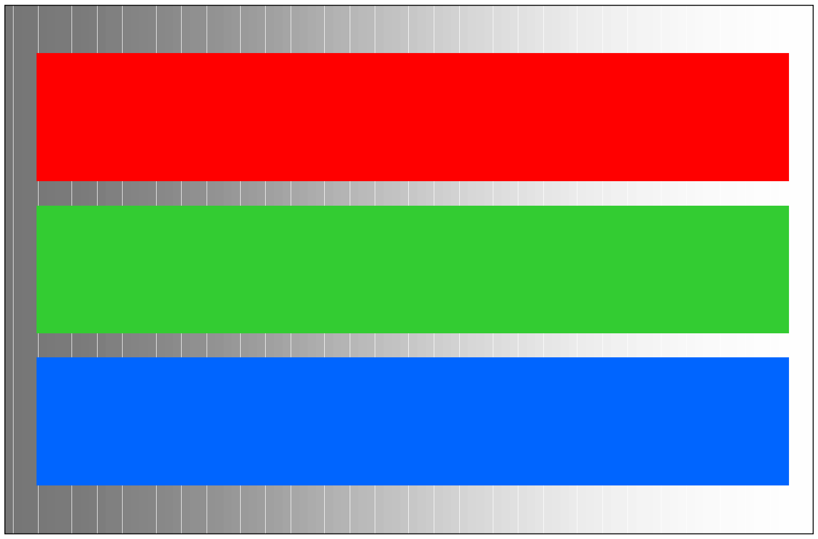
University of California, San Diego

San Diego Supercomputer Center



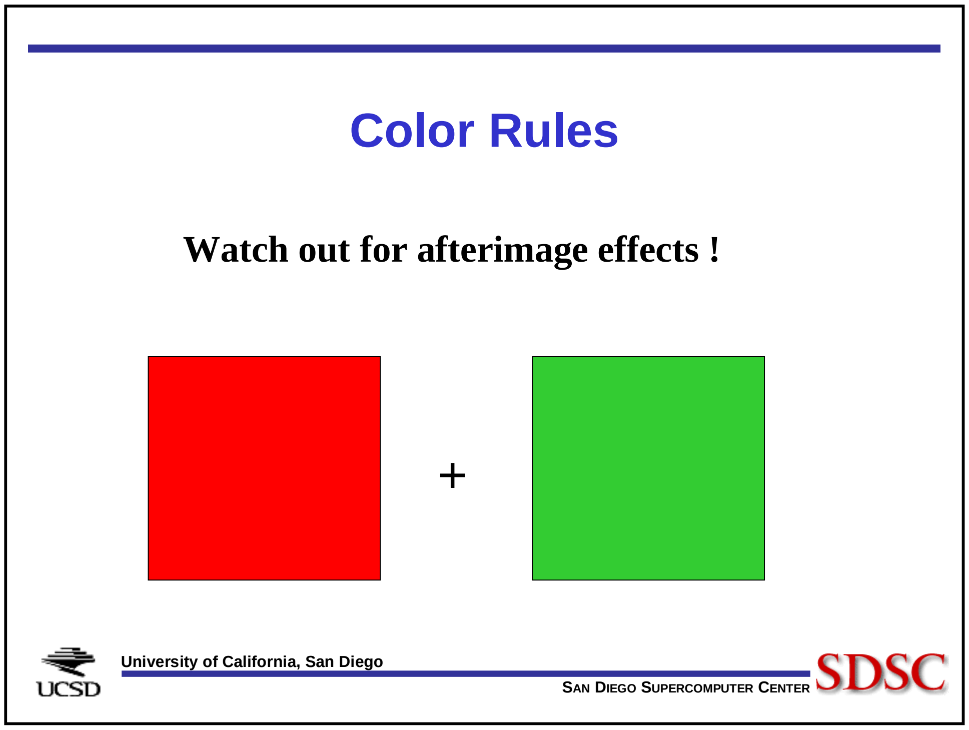
University of California, San Diego

San Diego Supercomputer Center SDSC



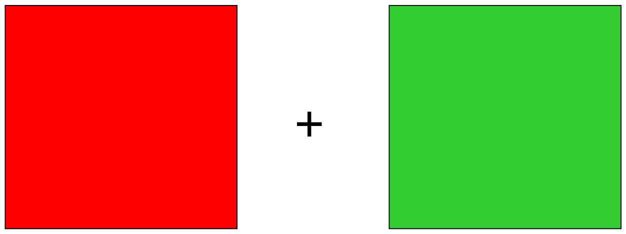
University of California, San Diego

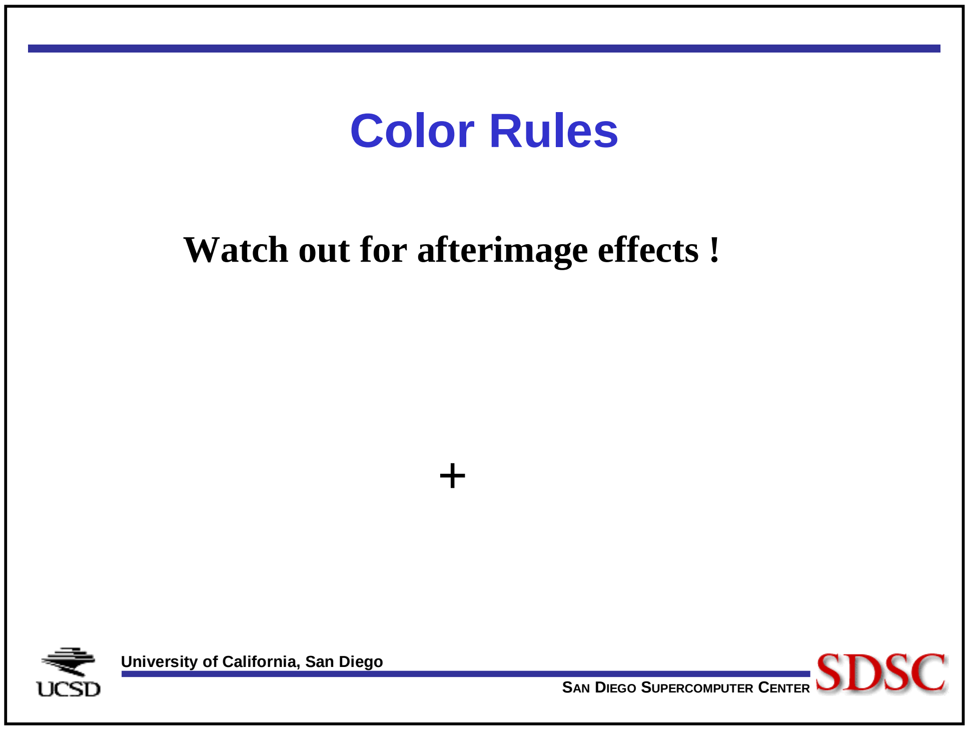
San Diego Supercomputer Center SDSC



Color Rules

Watch out for afterimage effects !



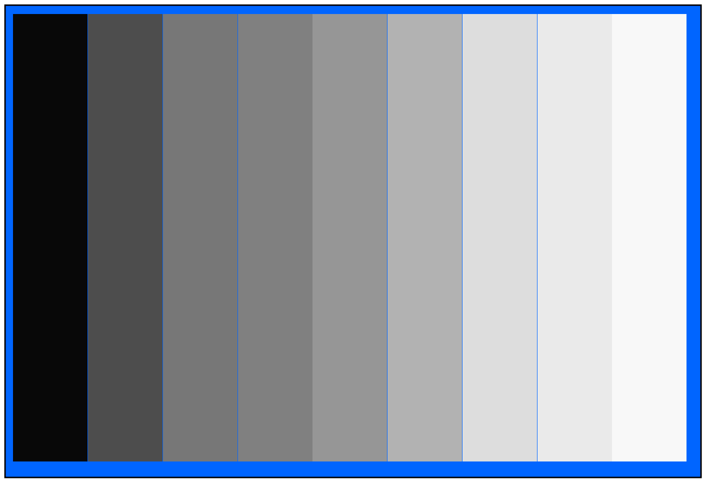


Color Rules

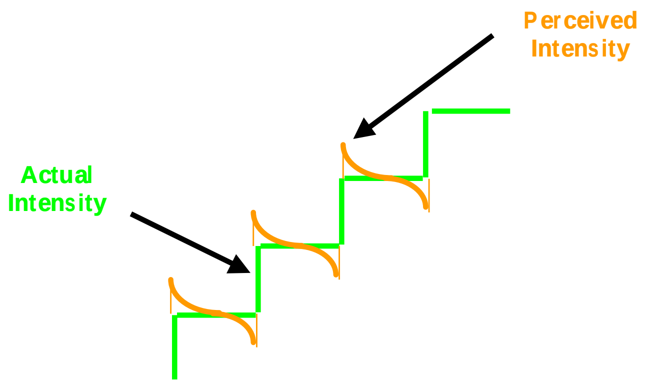
Watch out for afterimage effects !

+

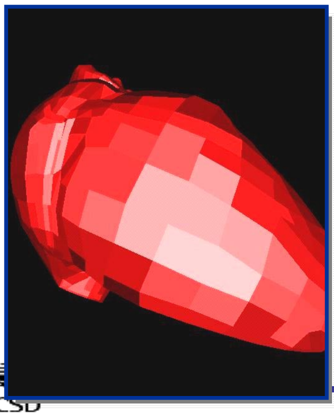
Beware of Mach Banding

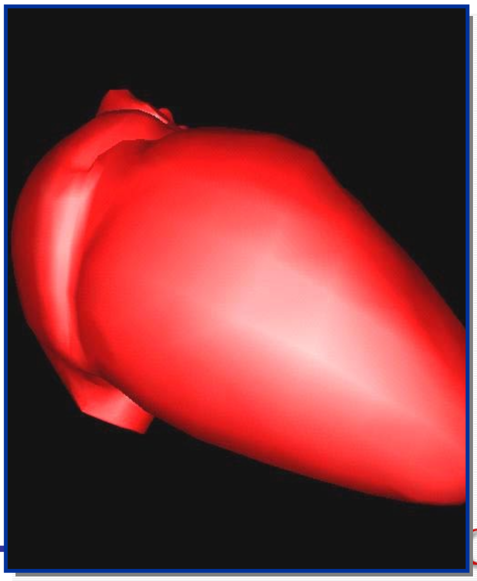


Beware of Mach Banding

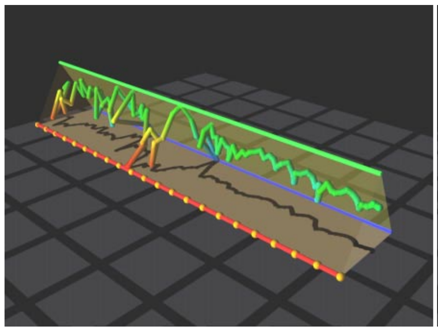


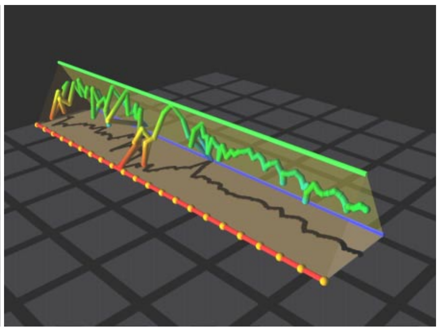
Beware of Mach Banding





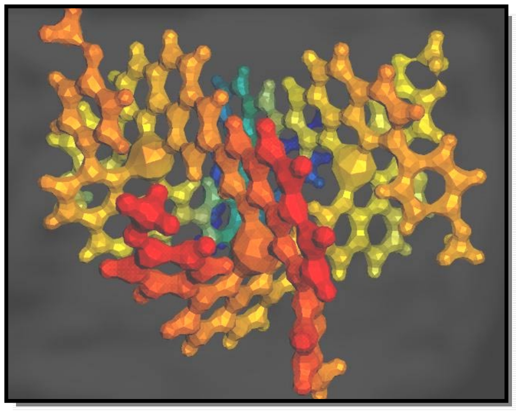
Cool Tricks: Stereopairs

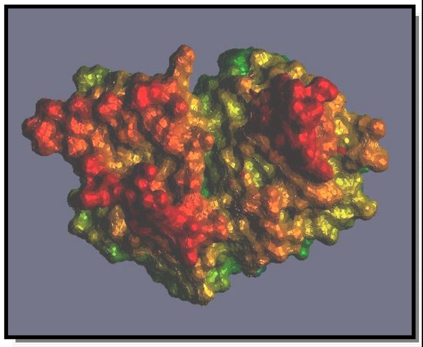




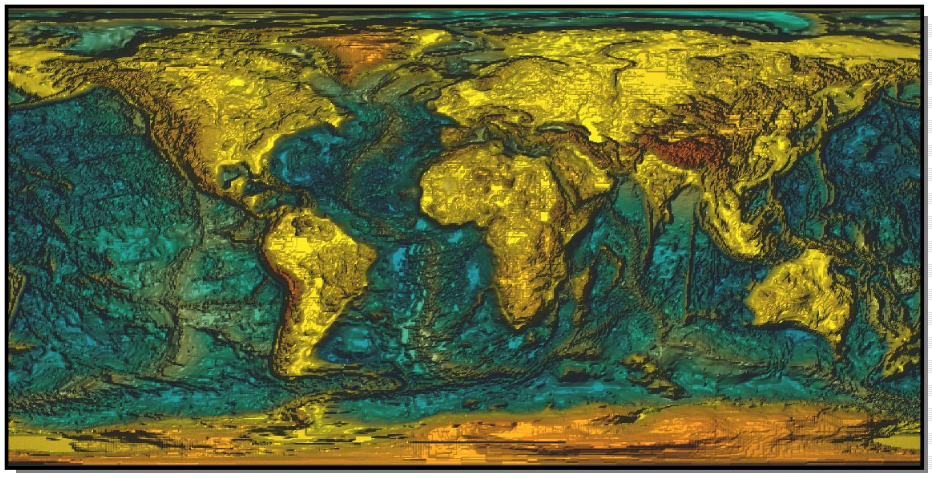
Left **Right**

Cool tricks: ChromaDepth





See: `http://www.sdsc.edu/~mjb/chromadepth`



Visualization Hardcopy

- Film recording
- Color printing
- Videotaping
- Manufacturing

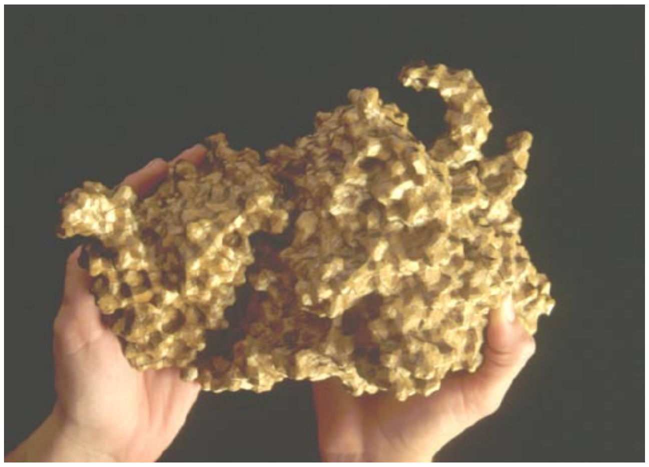
NTSC Videotaping for Visualization Hardcopy

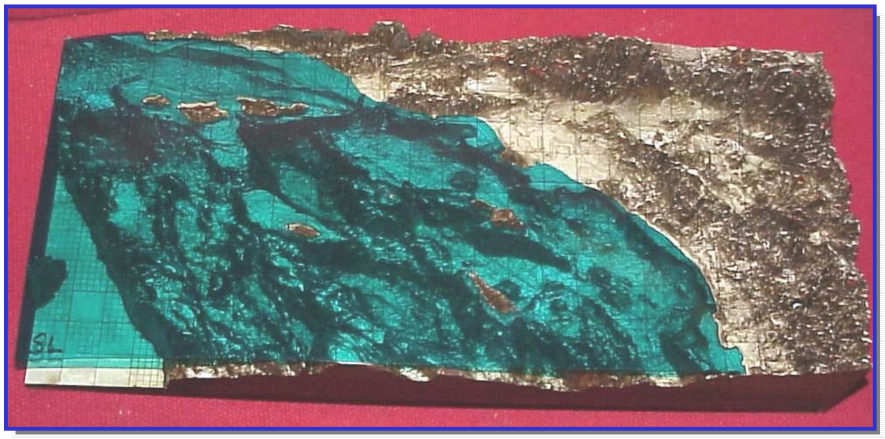
- **Resolution is 640x480, at best**
- **Interlaced**
- **Color bandwidth is given to, in order:**

1. Luminance	**267 cycles/scanline**
2. Blue-Orange	**96 cycles/scanline**
3. Purple-Green	**35 cycles/scanline**

- **Saturated (bright) colors don’t come out well**

Manufacturing for Visualization Hardcopy





The STL File Format

```
solid
  facet normal -0.62 -0.77 -0.12
    outer loop
      vertex 2.309218 0.639900 0.000000
      vertex 2.346300 0.609991 0.000000
      vertex 2.322692 0.621243 0.049971
    endloop
  endfacet
  facet normal -0.64 -0.76 -0.11
    outer loop
      vertex 2.292096 0.639900 0.100000
      vertex 2.309218 0.639900 0.000000
      vertex 2.322692 0.621243 0.049971
    endloop
  endfacet

     . . .

  facet normal  0.00  0.00  1.00
    outer loop
      vertex 2.568488 5.119200 4.500000
      vertex 2.346300 5.119200 4.500000
      vertex 2.559600 5.118925 4.500000
    endloop
  endfacet
endsolid
```

See: `http://cvp.sdsc.edu`

Summary

Scientific Visualization **has "no rules".**

Anything that turns *data* in the computer into *information* and *insight* in your brain is fair game.

Computer Graphics for Scientific Visualization

Mike Bailey

San Diego Supercomputer Center
University of California San Diego

`mjb@sdsc.edu`

`http://www.sdsc.edu/~mjb`

Input Devices for Computer Graphics

Mike Bailey

University of California San Diego
San Diego Supercomputer Center

`mjb@sdsc.edu`

Logical Input Device Types

- Choice
- Keyboard
- Valuators
- Locators

Choice

- Return a choice that has been made
- Examples: function keypad, button box, foot switch
- Often provide sensorial feedback: lights, clicks, feel, . . .

Button Box

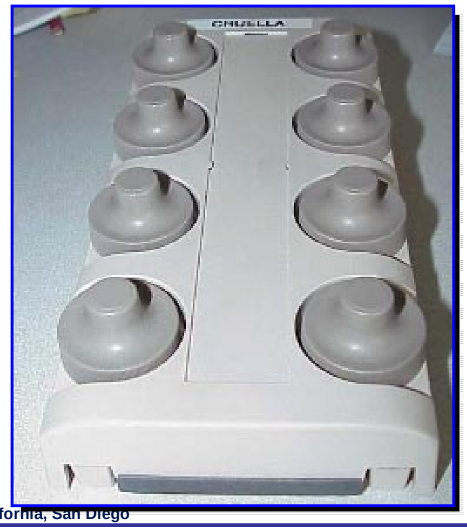
Keyboard

- Returns keys with specific meanings
- Letters, numbers, etc.

Valuators

- Return a value for something
- Example: knobs
- Can usually specify gain, minimum, and maximum
- All locators can also double as valuators

Dial Box



Locators

- Return the location of the screen cursor
- Examples: mouse, tablet, trackball, touchscreen
- Display-to-Input ratio

Locator Display-to-Input Ratio

DTI Ratio: the amount of cursor movement on the screen divided by the amount of hand movement

Large: good for speed

Small: Good for accuracy

"Sometimes called Gain"

Ways to Read an Input Device

- **Sampling: What is its input *right now* ?**
- **Event-based: Wait until the user does something**

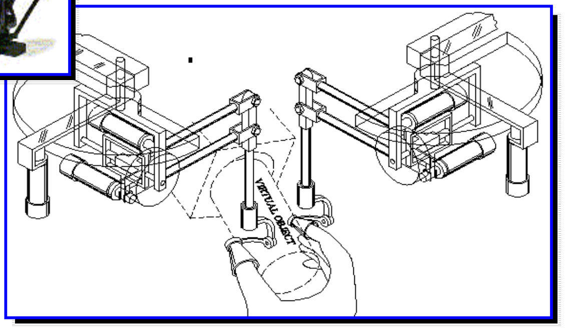
3D Input devices

- Read a 3D position
- Returns 3 numbers to the program: an (x,y,z) triple
- Some also return 3 more numbers to the program: three rotation angles
- Examples: digitizer, spaceball, glove

3D Input Devices

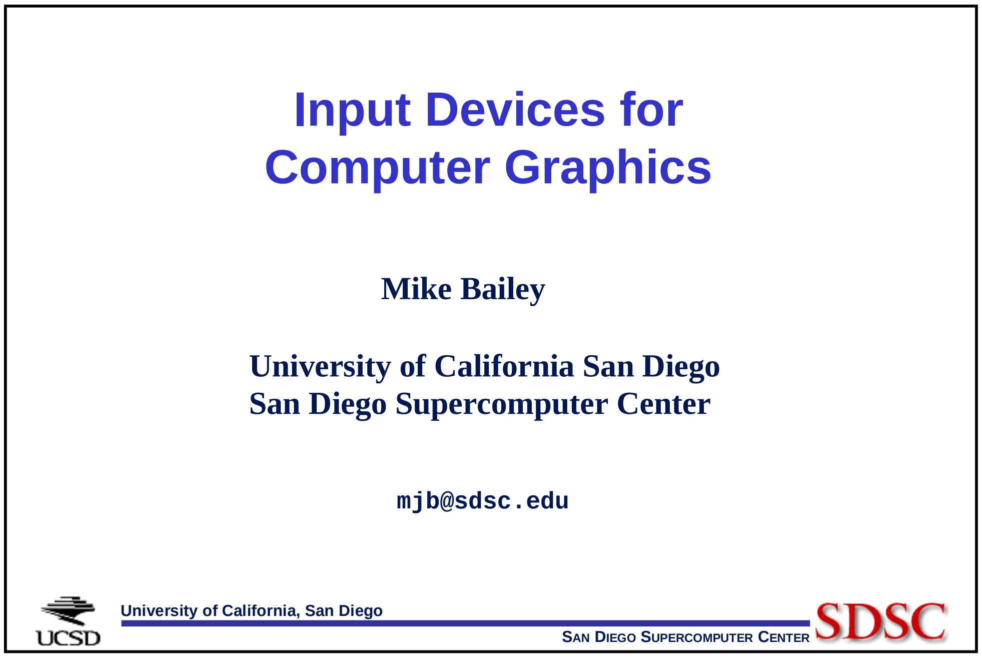
3D Input Devices

Force Feedback in 3D



Force Feedback in 3D

Force Feedback in 2D



Input Devices for Computer Graphics

Mike Bailey

University of California San Diego
San Diego Supercomputer Center

mjb@sdsc.edu

Computer Graphics on the World Wide Web

Mike Bailey

University of California San Diego
San Diego Supercomputer Center

`mjb@sdsc.edu`

Image Files

- GIF
- JPEG
- PNG

GIF Image Files

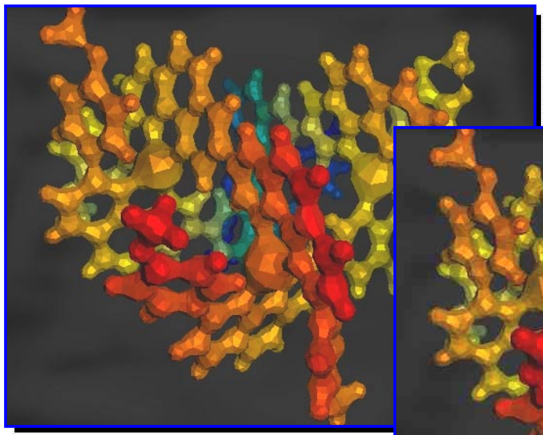
- Graphics Interchange Format
- Up to 8 bits with CLT
- Can be interleaved
- Can have a transparent background
- Can be compressed
- Can have multiple frames, an interframe delay time, and a repeat count

JPEG Image Files

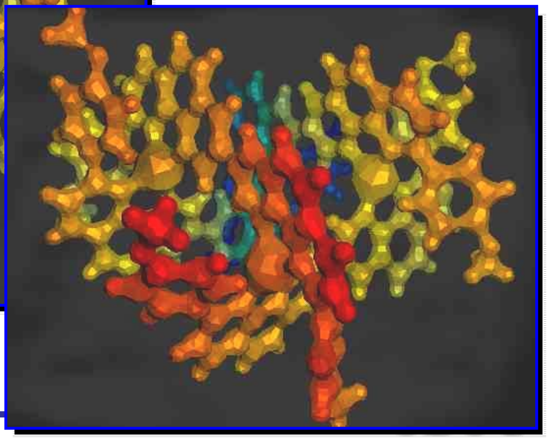
- Joint Photographic Experts Group
- Image can be 24-bit color
- Can be arbitrarily compressed
- `http://www.jpeg.org`
- The following example shows a 1080x852 24-bit image (uncompressed = 2,760,480 bytes)

JPEG Compression

174,628 Bytes

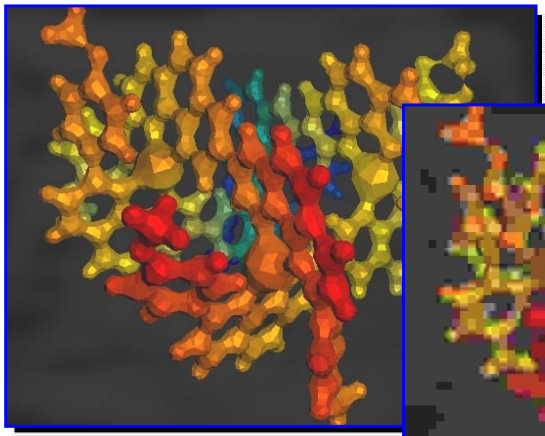


16,167 Bytes

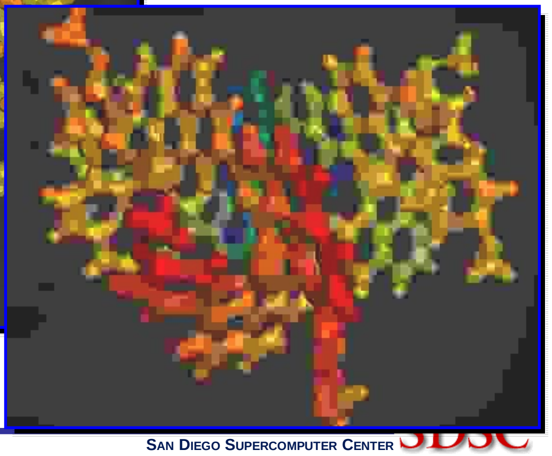


JPEG Compression

174,628 Bytes



5,311 Bytes

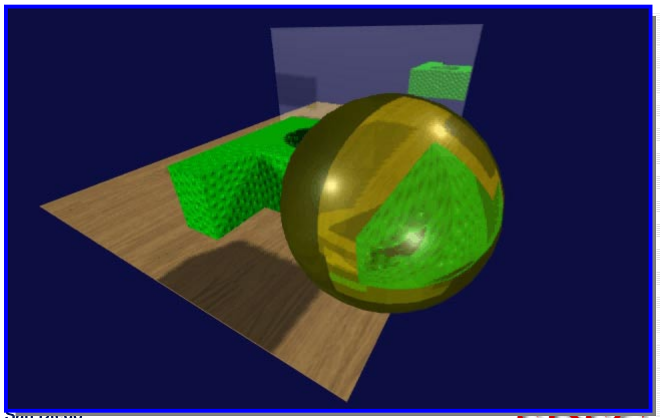


PNG

- Portable Network Graphics
- Designed to replace GIF
  - variable transparency
  - 2D interleaving
  - better compression
- `http://www.cdrom.com/pub/png`

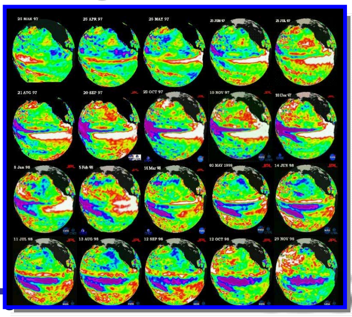
Animation Files

- Animated GIF
- MPEG
- Quicktime



Animated GIF

- GIF files can have multiple frames, an interframe delay time, and a repeat count
- One freeware package is WhirlGif: `www.danbbs.dk/~dino/whirlgif`



MPEG

- Moving Picture Experts Group
- Codes video and audio
- Highly compressed
- `www.mpeg.org`

Quicktime

- Product of Apple Computer
- Codes video and audio
- Highly compressed
- `www.apple.com/quicktime`

VRML

- Virtual Reality Modeling Language
- 3D scene coded as an ASCII file
- Scene content (ie, geometry) defined in nodes
- Node attributes given as fields and values (eg, size, appearance, lighting properties)

VRML Example

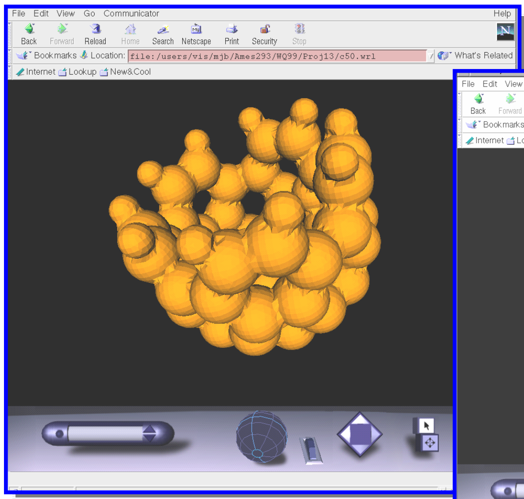
```
Cylinder
{}


Cylinder
{
  radius   2.0
  height   4.0
}
```

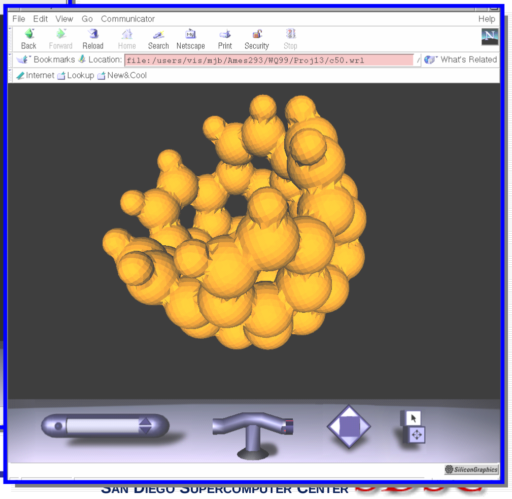
VRML Example

```
#VRML 2.0 utf8
Shape
{
  appearance
    Appearance
    {
      material
        Material { }
    }
  geometry
    Cylinder
    {
      radius  2.0
      height  4.0
    }
}
```

VRML on UNIX

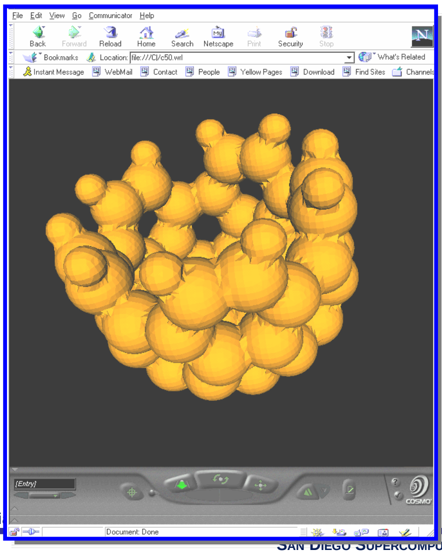

File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Stop
Bookmarks
What's Related
Internet Lookup New&Cool

Location: file:/users/vis/mjb/Ames293/WQ99/Proj13/c50.wrl


File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Stop
Bookmarks
What's Related
Internet Lookup New&Cool
SiliconGraphics

Location: file:/users/vis/mjb/Ames293/WQ99/Proj13/c50.wrl

VRML on Windows


File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Stop
Bookmarks
What's Related
Instant Message WebMail Contact People Yellow Pages Download Find Sites Channels
[Entry]
COSMO

Location: file:///C|/c50.wrl

Document: Done

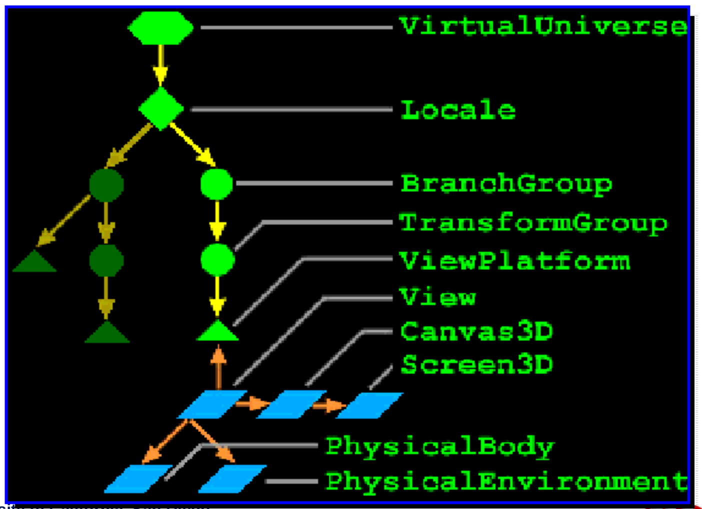
For More VRML Information, See:

`http://www.web3d.org`

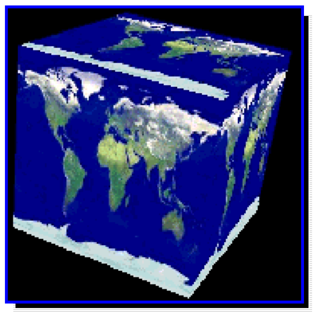
Java 3D

- 3D Graphics API for Java
- Primary mode is a 3D scene-graph display list
- Display list is highly optimized
- View model is separate from the scene model

Java 3D Scene Graph



Java 3D Examples



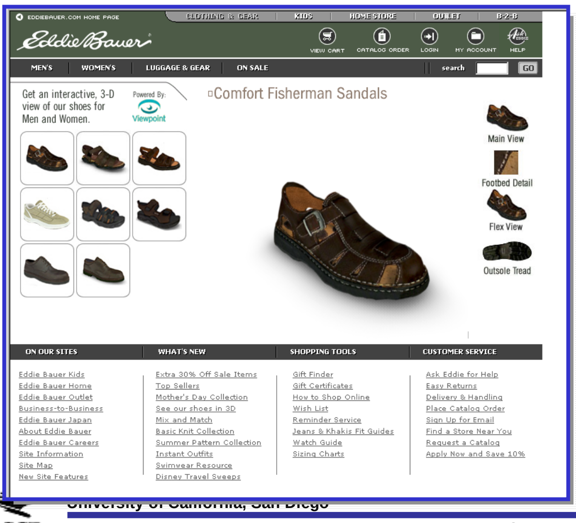
For More Java 3D Information, See:

`http://www.sun.com/desktop/java3d`

`htp://java.sun.com/products/java-media/3D`

`http://www.web3d.org`

Other 3D Web-based Standards



Viewpoint Data Labs plug-in

`www.viewpoint.com`

Other 3D Web-based Standards

Macromedia Shockwave 3D plug-in

`www.macromedia.com`

Computer Graphics on the World Wide Web

Mike Bailey

University of California San Diego
San Diego Supercomputer Center

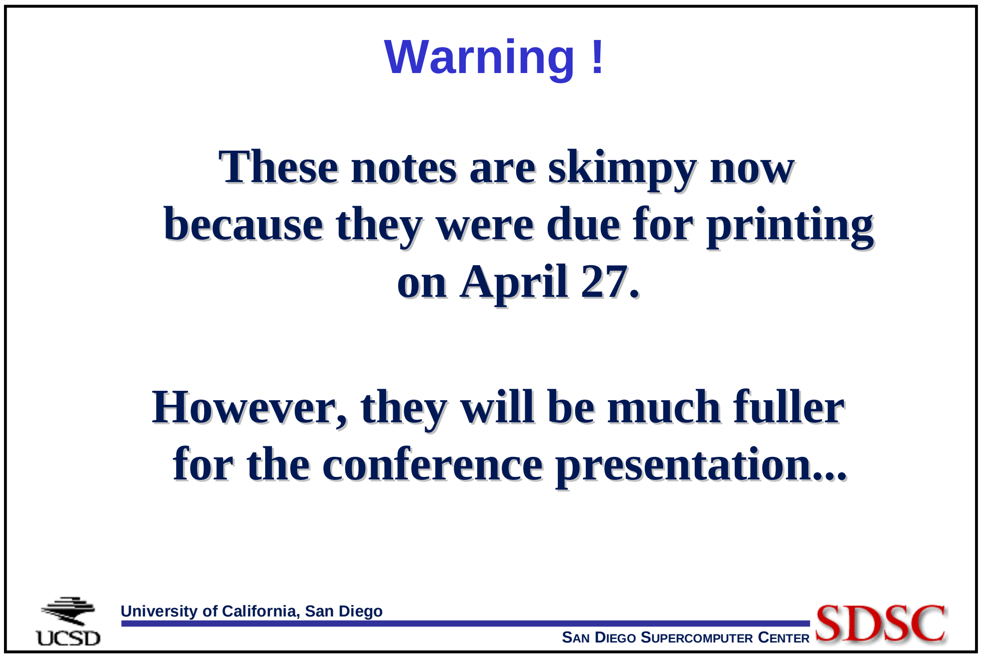
`mjb@sdsc.edu`

How To Attend a SIGGRAPH Conference

Mike Bailey

University of California San Diego
San Diego Supercomputer Center

mjb@sdsc.edu



Warning !

These notes are skimpy now because they were due for printing on April 27.

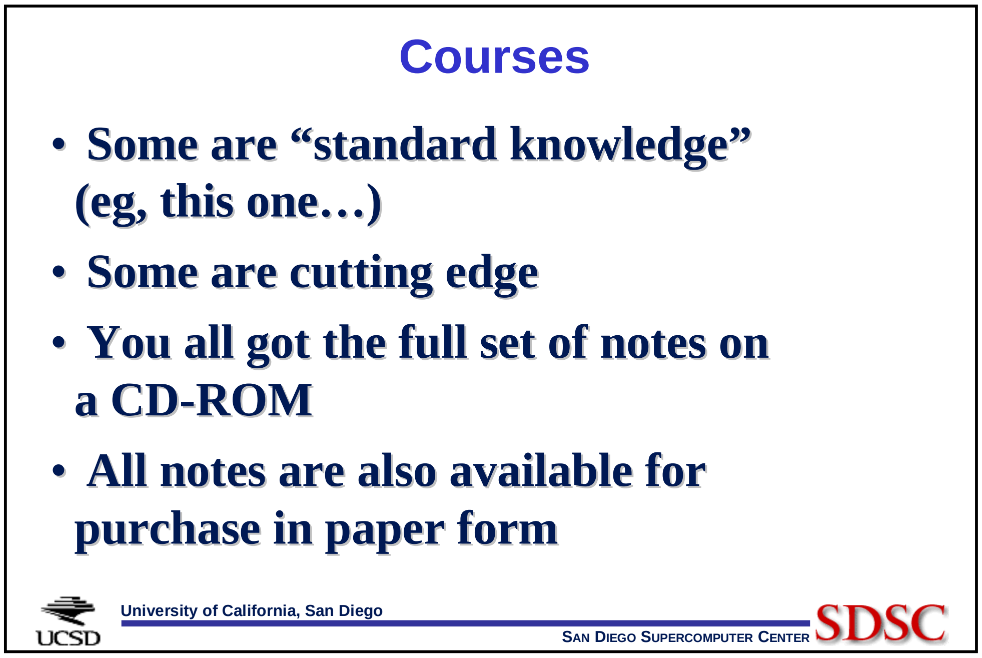
However, they will be much fuller for the conference presentation...

Papers

- Deep technical information
- Also printed in the proceedings and on a CD-ROM

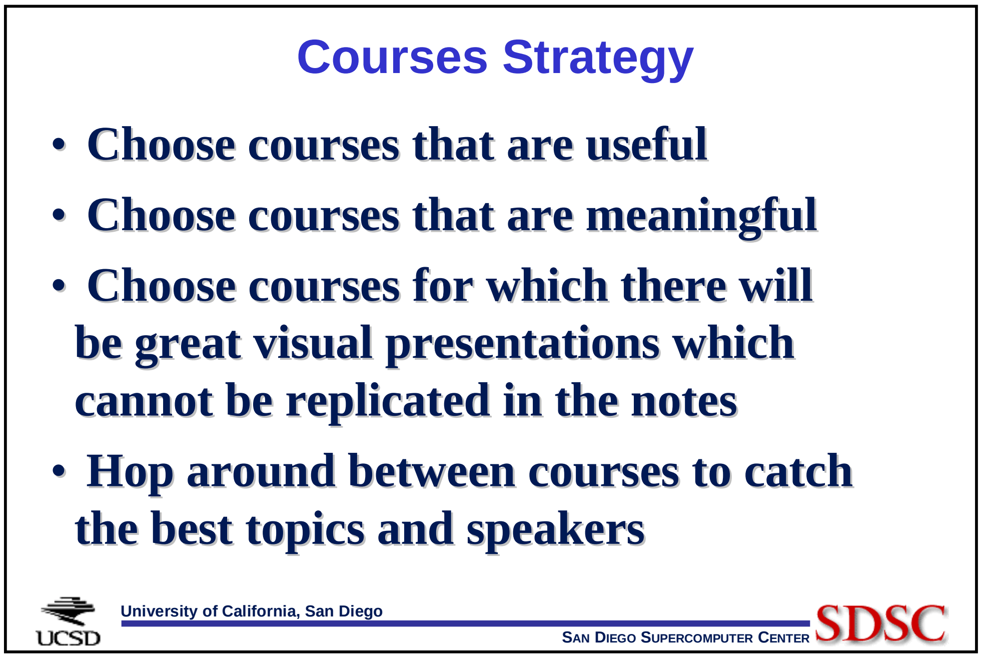
Panels

- Less formal than Papers
- Not archived anywhere
- Often controversial
- Sometimes confrontational



Courses

- Some are “standard knowledge” (eg, this one…)
- Some are cutting edge
- You all got the full set of notes on a CD-ROM
- All notes are also available for purchase in paper form



Courses Strategy

- Choose courses that are useful
- Choose courses that are meaningful
- Choose courses for which there will be great visual presentations which cannot be replicated in the notes
- Hop around between courses to catch the best topics and speakers

Exhibition

- ~200 vendors
- Many vendors time their hottest product releases for this week!

Exhibition Strategy

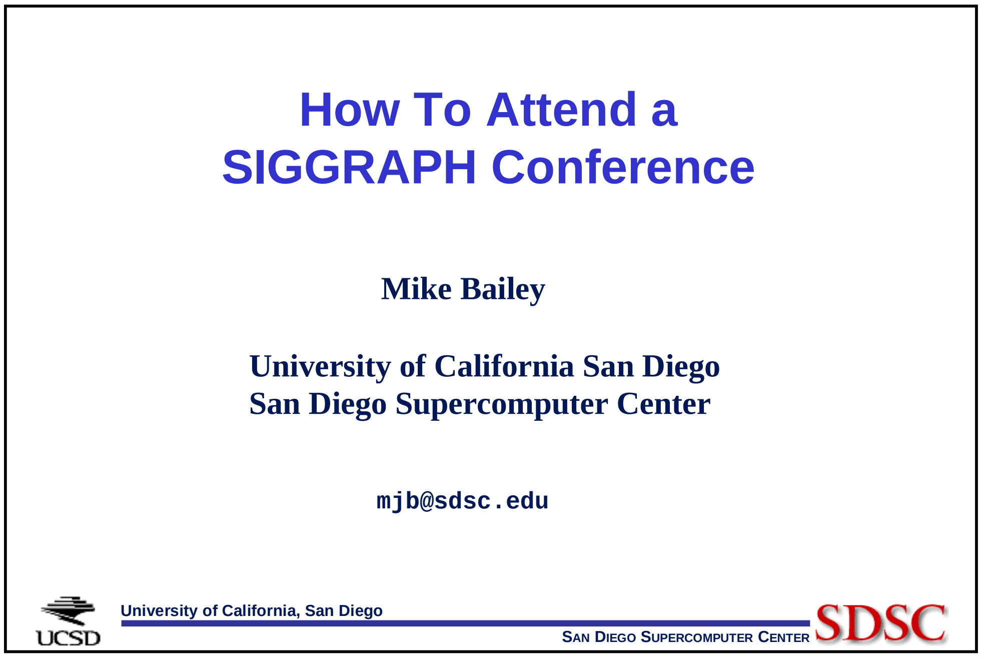
- Look at the list of vendors in the Final Program
- Make a list of the ones you *really* must see
- Sort the list by booth number
- Start at one end of the floor and work your way across

Exhibition Warning

The Exhibition closes on Thursday afternoon !!!

Electronic Theatre

- **Computer Graphics's greatest animation hits for the past year**
- **It is considered *cool* to see it early in the week**
- **Watch for whose piece goes last**



How To Attend a SIGGRAPH Conference

Mike Bailey

University of California San Diego
San Diego Supercomputer Center

`mjb@sdsc.edu`

Where to Find More Information about Computer Graphics and Scientific Visualization

Mike Bailey
San Diego Supercomputer Center
University of California, San Diego

1. References

1.1 General

SIGGRAPH Online Bibliography Database:
`http://www.siggraph.org/publications/bibliography`

F. S. Hill, *Computer Graphics Using OpenGL*, Prentice Hall, 2001.

Edward Angel, *Interactive Computer Graphics: A Top-down Approach with OpenGL*, Addison-Wesley, 1999.

Olin Lathrop, *The Way Computer Graphics Works*, John Wiley & Sons, 1997.

Andrew Glassner, *Graphics Gems*, Academic Press, 1990.

James Arvo, *Graphics Gems 2*, Academic Press, 1991.

David Kirk, *Graphics Gems 3*, Academic Press, 1992.

Paul Heckbert, *Graphics Gems 4*, Academic Press, 1994.

Alan Paeth, *Graphics Gems 5*, Academic Press, 1995.

Jim Blinn, *A Trip Down the Graphics Pipeline*, Morgan Kaufmann, 1996.

Jim Blinn, *Dirty Pixels*, Morgan Kaufmann, 1998.

David Rogers, *Procedural Elements for Computer Graphics*, McGraw-Hill, 1997.

SIGGRAPH Conference Final program.

1.2 Math and Geometry

Jean Gallier, *Curves and Surfaces in Geometric Modeling*, Morgan Kaufmann, 2000.

Walter Taylor, *The Geometry of Computer Graphics*, Wadsworth & Brooks/Cole, 1992.

Gerald Farin, *Curves and Surfaces for Computer Aided Geometric Design*, Academic Press, 1996.

Gerald Farin and Dianne Hansford, *The Geometry Toolbox for Graphics and Modeling*, AK Peters, 1998.

Barrett O'Neil, *Elementary Differential Geometry*, Academic Press, 1997.

Joseph O'Rourke, *Computational Geometry in C*, Cambridge University Press, 1996.

Christopher Hoffman, *Geometric & Solid Modeling*, Morgan Kaufmann, 1989.

Michael Mortenson, *Geometric Modeling*, John Wiley & Sons, 1985.

I.D. Faux and M.J. Pratt, *Computational Geometry for Design and Manufacture*, Ellis-Horwood, 1979.

Eric Stollnitz, Tony DeRose, and David Salesin, *Wavelets for Computer Graphics*, Morgan-Kaufmann, 1996.

Ronen Barzel, *Physically-Based Modeling for Computer Graphics*, Academic Press, 1992.

David Rogers and J. Alan Adams, *Mathematical Elements for Computer Graphics*, McGraw-Hill, 1989.

John Snyder, *Generative Modeling for Computer Graphics and Computer Aided Design*, Academic Press, 1992.

1.3 Scientific Visualization

Chandrajit Bajaj, *Data Visualization Techniques*, John Wiley & Sons, 1999.

Min Chen, Arie Kaufman, and Roni Yagel, *Volume Graphics*, Springer-Verlag, 2000.

Will Schroeder, Ken Martin, and Bill Lorensen, *The Visualization Toolkit*, Prentice-Hall, 1998.

Greg Nielson, Hans Hagen, and Heinrich Müller, *Scientific Visualization: Overviews, Methodologies, Techniques,* IEEE Computer Society Press, 1997.

Lenny Lipton, *The CrystalEyes Handbook*, StereoGraphics Corporation, 1991.

Brand Fortner, *The Data Handbook: A Guide to Understanding the Organization and Visualization of Technical Data*, Spyglass, 1992.

William Kaufmann and Larry Smarr, *Supercomputing and the Transformation of Science,* Scientific American Library, 1993.

Robert Wolff and Larry Yaeger, *Visualization of Natural Phenomena*, Springer-Verlag, 1993.

David McAllister, *Stereo Computer Graphics and Other True 3D Technologies,* Princeton University Press, 1993.

Peter Keller and Mary Keller, *Visual Cues: Practical Data Visualization*, IEEE Press, 1993.

1.4 Color and Perception

Roy Hall, *Illumination and Color in Computer Generated Imagery*, Springer-Verlag, 1989.

David Travis, *Effective Color Displays*, Academic Press, 1991.

L.G. Thorell and W.J. Smith, *Using Computer Color Effectively*, Prentice Hall, 1990.

Edward Tufte, *The Visual Display of Quantitative Information*, Graphics Press, 1983.

Edward Tufte, *Envisioning Information*, Graphics Press, 1990.

Edward Tufte, *Visual Explanations*, Graphics Press, 1997.

Howard Resnikoff, *The Illusion of Reality*, Springer-Verlag, 1989.

1.5 Rendering

Andrew Glassner, *Principles of Digital Image Synthesis*, Morgan Kaufmann, 1995.

Andrew Glassner, *An Introduction to Ray Tracing*, Academic Press, 1989.

Rosalee Wolfe, *3D Graphics: A Visual Approach*, Oxford Press.

Steve Upstill, *The RenderMan Companion,* Addison-Wesley, 1990.

Tony Apodaca and Larry Gritz, *Advanced RenderMan: Creating CGI for Motion Pictures*, Morgan Kaufmann, 1999.

Ken Joy et al, *Image Synthesis*, IEEE Computer Society Press, 1988.

1.6 Images

David Ebert et al, *Texturing and Modeling,* Academic Press, 1998.

Ron Brinkman, *The Art and Science of Digital Compositing*, Morgan Kaufmann, 1999.

John Miano, *Compressed Image File Formats*, Addison-Wesley, 1999.

1.7 Animation

Alan Watt and Mark Watt, *Advanced Animation and Rendering Techniques*, Addison-Wesley, 1998.

Nadia Magnenat Thalmann and Daniel Thalmann, *Interactive Computer Animation*, Prentice-Hall, 1996.

Philip Hayward and Tana Wollen, *Future Visions: New Technologies of the Screen*, Indiana University Press, 1993.

1.8 Virtual Reality

John Vince, *Virtual Reality Systems*, Addison-Wesley, 1995.

1.9 The Web

Andrea Ames, David Nadeau, John Moreland, *The VRML 2.0 Sourcebook*, John Wiley & Sons, 1997.

Bruce Eckel, *Thinking in Java*, Prentice-Hall, 1998.

David Flanagan, *Java in a Nutshell*, O'Reilly & Associates, 1996.

David Flanagan, *Java Examples in a Nutshell*, O'Reilly & Associates, 1997.

Henry Sowizral, Kevin Rushforth, and Michael Deering, *The Java 3D API Specification*, Addison-Wesley, 1998.

1.10 Miscellaneous

OpenGL 1.2 Reference Manual, Addison-Wesley, 2000.

OpenGL 1.2 Programming Guide, Addison-Wesley, 2000.

Andrew Glassner, *Recreational Computer Graphics*, Morgan Kaufmann, 1999.

Anne Spalter, *The Computer in the Visual Arts*, Addison-Wesley, 1999.

Jef Raskin, *The Humane Interface*, Addison-Wesley, 2000.

Ben Shneiderman, *Designing the User Interface*, Addison-Wesley, 1997.

Clark Dodsworth, *Digital Illusion*, Addison-Wesley, 1997.

Isaac Victor Kerlow, *The Art of 3-D: Computer Animation and Imaging*, 2000.

Isaac Victor Kerlow and Judson Rosebush, *Computer Graphics for Designers and Artists*, Van Nostrand Reinhold, 1986.

William Press, Saul Teukolsky, William Vetterling, and Brian Flannery, *Numerical Recipes in C*, Second Edition, Cambridge University Press, 1997.

2. Periodicals

Computer Graphics and Applications: published by IEEE
(`http://www.computer.org`, 714-821-8380)

Computer Graphics World: published by Pennwell
(`http://www.cgw.com`, 603-891-0123)

Journal of Graphics Tools: published by A.K. Peters
(`http://www akpeters.com`, 617-235-2210)

Computer Graphics Quarterly: published by ACM SIGGRAPH
(`http://www.siggraph.org`, 212-869-7440)

Computers in Science and Engineering (used to be *Computers in Physics*): published by the American Institute of Physics

Transactions on Visualization and Computer Graphics: published by IEEE
(`http://www.computer.org`, 714-821-8380)

Transactions on Graphics: published by ACM
(`http://www.acm.org`, 212-869-7440)

Cinefex
(`http://www.cinefex.com`, 909-781-1917)

3. Professional organizations

ACM ....Association for Computing Machinery
`http://www.acm.org`, 212-869-7440

SIGGRAPH....ACM Special Interest Group on Computer Graphics
`http://www.siggraph.org`, 212-869-7440

SIGCHI	ACM Special Interest Group on Computer-Human Interfaces `http://www.acm.org/sigchi`, 212-869-7440
IEEE	Institute of Electrical and Electronic Engineers `http://www.computer.org`, 202-371-0101
NAB	National Association of Broadcasters `http://www.nab.org`, 800-521-8624
ASME	American Society of Mechanical Engineers `http://www.asme.org`, 800-THE-ASME

4. Conferences

ACM SIGGRAPH:
2002: San Antonio
July 21-26
`http://www.siggraph.org/s2002`
2003: San Diego

IEEE Visualization:
2001: San Diego, CA
October 21-26
`http://vis.computer.org`
2002: Boston, MA
2003: Seattle, WA

NAB:
Las Vegas, NV
`http://www.nab.org`

ACM SIGCHI:
2002: Minneapolis, MN – April 20-25
`http://www.acm.org/sigchi`

ACM SIGARCH / IEEE Supercomputing:
2001: Denver, CO – November 10-16
`http://www.supercomp.org`

5. Graphics Performance Characterization

The GPC web site tabulates graphics display speeds for a variety of vendors' workstation products. To get the information, ping:

`http://www.specbench.org/gpc`

Glossary of Introductory Computer Graphics Terms

This glossary was taken from the Glindex chapter (combined glossary and index) of the introductory book *The Way Computer Graphics Works*, by Olin Lathrop, published by John Wiley and Sons, 1997, ISBN 0-471-13040-0.

Copyright Notice

An on-line version of this glossary is available at http://www.cognivis.com/book/glossary.htm.

Click on any of the following letters to jump to the section for that letter.

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

2D

Two dimensional.

2D display controller

A type of display controller that is intended for 2D operations, like windows, popup menus, text, etc. This kind of display controller can usually not interpolate color values within a primitive, and therefore lacks support for 3D drawing.

2 1/2 D display controller

A 2D display controller that is able to perform the 2D operations required to display 3D primitives. This usually means color interpolation and Z buffering.

3D

Three dimensional.

3D display controller

A type of display controller that fully supports 3D operations and primitives. At a minimum, such a display controller can accept a 3D triangle, apply a 3D transform, perform the apparent color determination at the vertices, project it onto the 2D bitmap, and draw it with hidden surfaces suppressed. Today’s 3D display controllers use the Z buffer algorithm for hidden surface suppression.

3D texture

A texture map that is a function of three variables. This is also called a "solid texture", because the texture map is a volume. Solid textures have been used in diffuse color texture mapping to simulate things like marble and wood grain. Sometimes specifying the color in a volume is simpler than specifying it on the surface of an object, especially if the object has a complex shape.

3D transform

The act of converting the coordinates of a point, vector, etc., from one coordinate space to another. The term can also refer collectively to the numbers used to describe the mapping of one space on another, which are also called a "3D transformation matrix."

3D transformation matrix

See "3D transform."

- - - - A - - - -

adaptive space subdivision

Space subdivision refers to the process of breaking up the scene space into many small regions. Adaptive means this is only done where and when needed, instead of in a fixed way up front. Adaptive space subdivision is often used by ray tracers. Octrees and BSP trees are examples of subdivision algorithms that can be adaptive.

addition, vector

See "vector addition."

aliasing

The phenomenon that makes smooth lines and edges appear stair stepped or jagged. Aliasing is also called the "jaggies".

aliasing, temporal

See "temporal aliasing."

alpha buffer

The collective name for the alpha values for every pixel of an image or bitmap.

alpha buffered rendering

Using an alpha buffer for rendering, as opposed to for image compositing or matting. Alpha buffered rendering implies the ability to render semi-transparent primitives.

alpha buffering

The process of rendering or compositing images using an alpha buffer. An alpha buffer supplies an opacity fraction for every pixel.

alpha value

The alpha buffer value for a single pixel. An alpha value is a value indicating the pixels opacity. Zero usually represents totally transparent (invisible) and the maximum value represents completely opaque. Alpha values are commonly represented in 8 bits, in which case transparent to opaque ranges from 0 to 255.

ambient light

A light source that shines equally on everything. This is a hack used to give some illumination to areas that are not in direct view of any light source. In the real world, such areas are illuminated indirectly by light bouncing off other objects. Ambient illumination is commonly used except in radiosity rendering, because radiosity actually computes light bouncing between objects.

AND operator

A constructive solid geometry (CSG) modeling operation on two objects. The resulting object exists only where both input objects existed. This operation is also call INTERSECTION.

animation

Any method that can make an image appear to change over time. In computer graphics, this is done by showing many still images in rapid succession. This produces the illusion of a moving, or animated, image.

animation, inverse dynamics

See "inverse dynamics."

animation, inverse kinematics

See "inverse kinematics."

animation, key frame

See "key frame animation."

animation, parametric

See "parametric animation."

anti-aliasing

The process of reducing aliasing, or jaggies, in creating an image.

anti-aliasing filter, box

A box filter used in anti-aliasing averages all the samples of a high resolution image within each resulting pixel. All the samples are weighted equally over a rectangular region, usually the resulting anti-aliased pixel. Box filtering provides fair to medium quality results, but is much less complex than higher quality methods.

anti-aliasing filter, good quality

A good quality anti-aliasing filter blends values from a high resolution image such that samples near the resulting pixel center are weighted more than others. Sample weights smoothly approach zero at a distance of about 1¼ pixels.

anti-aliasing filter, simple

See "anti-aliasing filter, box."

apparent color determination

See "color determination."

apparent surface orientation

The orientation (which direction it’s facing) a surface appears to have in an image. This is controlled by the shading normal vector, which is not necessarily the same as the normal vector of the primitive as it’s actually drawn (the geometric normal vector).

- - - - B - - - -

B spline

A particular type of spline, the mathematical details of which are beyond the scope of this book.

back end

See "video back end".

basis vector

A vector that defines one axis of a coordinate system. Three basis vectors, one each for the X, Y and Z axis are needed to define a 3D coordinate system. A basis vector indicates the length and direction of a +1 increment in its axis.

beam current

The current of an electron beam in a cathode ray tube. The current is the number of electrons per unit time, which is usually measured in milliamperes. A higher beam current produces a brighter phosphor dot.

beta spline

A particular type of spline, the mathematical details of which are beyond the scope of this book.

bi-cubic patch

A particular type of surface patch. Bi-cubic surface patches can have curved edges, as opposed to polygons, which always have straight edges. The mathematical details of bi-cubic patches are beyond the scope of this book.

bi-quadratic patch

A particular type of surface patch. Bi-quadratic surface patches can have curved edges, as opposed to polygons, which always have straight edges. The mathematical details of bi-quadratic patches are beyond the scope of this book.

bi-linear interpolation

Interpolation is the process of determining plausible in-between values, given explicit values at particular points. Linear means that the values fall along a line from one known point to the next. This means the value changes a fixed amount for a fixed-sized step. Bi-linear means this process is carried out in two dimensions. In computer graphics, bi-linear interpolation is often applied to find color values at the interior pixels of a primitive. The apparent color values are computed explicitly at the vertices of a polygon and are bi-linearly interpolated in the polygon’s interior. Bi-linear interpolation of pixel color values is also called Gouraud shading.

binary space partition

See "BSP tree."

bitmap

The collective name for all the stored pixels in a display controller. The bitmap is also the interface between the display controller’s drawing front end and its video back end. The term bitmap is also used in general to refer to any 2D array of pixels.

blobbies

A name sometimes applied to potential functions used in modeling objects.

blue screen

A background often used for photographs or video that are to be matted, or composited, over other images. A blue background is almost universally used in chroma keying video compositing.

boolean operator

A mathematical operator that works on true or false values. These are also called logical operators. In computer graphics, constructive solid geometry (CSG) operators may also be called boolean operators. Some common CSG operators are AND, OR, NOT, and XOR.

bounding volume

A volume that encloses multiple 3D primitives. If the bounding volume doesn’t intersect an object of interest (such as a ray in ray tracing), then the objects within the bounding volume are guaranteed to also not intersect the object, eliminating the need to check explicitly.

box filter

See "anti-aliasing filter, box."

BSP tree

A hierarchical method of subdividing a volume. BSP stands for "binary space partition." In this method, the volume is originally represented as one whole. If more detail is needed, the volume is subdivided into two volumes. Each of these are further subdivided into two volumes if more detail is needed. The process is repeated until the desired level of detail is achieved, or an arbitrary subdivision limit is reached.

BSP tree, regular

A special form of BSP tree where all the volume elements are rectangular solids that are always subdivided exactly in half along one of their three major axes.

bump mapping

A form of texture mapping where the texture supplies small perturbations of the shading normal vector.

- - - - C - - - -

calligraphic

An early type of computer graphics display that could only draw lines, not filled in areas. Calligraphic displays are rarely used today, and have mostly been replaced by raster displays.

camera point

See "eye point."

cathode ray tube

A type of vacuum tube that is commonly used as a computer graphics output device. A thin beam of electrons is shot at a spot on the inside of the tube’s face. The inside of the face is coated with phosphors that emit light when the beam hits them. The beam is swept in a raster pattern to hit every spot on the screen. The beam current is modulated to make light and dark areas on the phosphors, forming an image. Cathode ray tube is usually abbreviated as CRT.

chroma keying

An image compositing technique commonly used on video signals. An overlay video signal is selected instead of a background video signal whenever the overlay isn’t a particular preset hue, usually blue. Action shot in front of a blue screen can thereby appear on top of the background signal.

circular reasoning

See "reasoning, circular."

color determination

The process of figuring out what the apparent color of a particular point on a particular primitive is. This process is used to answer the question "What color is the object at this pixel?"

color, diffuse

See "diffuse color."

color, emissive

See "emissive color."

color, specular

See "specular color."

color index value

The input value to a color lookup table (LUT) of a display controller’s video back end operating in pseudo color mode. Color index values are also referred to as pseudo colors.

color lookup table

A table of color values in a display controller’s video back end. In pseudo color mode, it translates the pseudo color values into RGB color values. In true color mode it becomes three separate tables, one for each red, green, and blue component. It then translates the red, green, and blue pixel component values to the final displayed red, green and blue component values. The color lookup table is usually just called the LUT.

color purity

The degree to which a color CRT can display just one of its three red, green, or blue primary colors without displaying any portion of the other two. This is a measure of how much each electron gun can only hit the phosphor dots of its color.

color space

A scheme for describing different shades or colors. The RGB color space defines a shade as a mixture of specific quantities of red, green, and blue.

color space, RGB

See "RGB."

color space, IHS

See "IHS."

color wheel

A circular diagram of colors. The hue varies as the angle within the disc. Saturation increases from the center outward. The entire disc is usually shown at the same intensity.

compositing

The process of combining two images to yield a resulting, or composite, image. Alpha buffering is a common compositing technique in computer graphics.

compression

As used in this book, the process of encoding an image that contains redundant information such that it requires less storage. Runlength and LZW encoding are examples of lossless compression techniques. JPEG and MPEG are examples of lossy compression techniques.

compression, JPEG

See "JPEG."

compression, lossless

See "lossless compression."

compression, lossy

See "lossy compression."

compression, LZW

See "LZW compression."

compression, MPEG

See "MPEG."

compression, runlength encoding

See "runlength encoding."

concave

A property of a polygon that has at least one vertex bulge inward instead of outward. See the text for a more rigorous definition.

constraint

A rule of an inverse kinematics or inverse dynamics animation system that must be adhered to in solving the motion of all the objects. For example, a constraint in animating a walking human might be "the lower end of the leg always remains joined to the foot at the ankle."

constructive solid geometry

A modeling technique where complex shapes are built by combinations of simple shapes. Shapes are combined with boolean operators such as AND, OR, XOR, MINUS, and NOT. For example, a box with a hole thru it could be defined as box minus cylinder.

control point

A point used in defining a spline.

convergence

The degree to which all three electron beams of a color CRT meet at the same spot on the screen. A poorly converged CRT will show the red, green, and blue components of the image slightly offset from each other. This makes the image look blurry and produces color fringes around the edges of objects.

convex

A property of a polygon that bulges outward at all vertices. See the text for a more rigorous definition.

convolution

A mathematical operation that applies a weighted average defined by one function onto another function. This is a very loose definition. A rigorous detailed one is beyond the scope of this book. Anti-aliasing is often done with convolutions. coordinate space Error! Bookmark not defined. A reference frame that defines numerical values, or coordinates, over an area (2D) or volume (3D).

cross product

A mathematical operation performed on two vectors, yielding a third vector. The third vector is always at right angles to the first two.

CRT

See "cathode ray tube."

CSG

See "constructive solid geometry."

CSG operator

See "boolean operator."

CSG AND operator

See "AND operator."

CSG EXCLUSIVE OR operator

See "XOR operator."

CSG INTERSECTION operator

See "AND operator."

CSG MINUS operator

See "MINUS operator."

CSG NOT operator

See "NOT operator."

CSG OR operator

See "OR operator."

CSG UNION operator

See "OR operator."

CSG XOR operator

See "XOR operator."

curved patch

A primitive used to model a small piece, or patch, of a surface. A curved patch, as opposed to a polygon, can have edges that are not straight.

- - - - D - - - -

de-Gauss

An action performed on a CRT monitor to de-magnetize the CRT and any material near it. CRTs are very sensitive to external magnetic fields. Such fields can cause alignment, convergence, purity, and image distortion problems. Most CRT monitors automatically perform de-Gaussing when they are first switched on.

deflection yoke

Coils mounted on a CRT that are used to steer the electron beam.

depth buffer

See "Z buffer."

depth buffer rendering

See "Z buffer rendering."

depth value

See "Z value."

diffuse color

An object surface property defined in the Phong lighting model. An object’s diffuse color is reflected equally in all directions. Visually, this is the not-shiny or dull color.

diffuse reflection

The light reflected from an object that is reflected equally in all directions. These are the kind of inter- object reflections that are modeled well by radiosity.

direct evaluation (of object color)

The process of determining the apparent color of an object at a particular point by direct evaluation of the lighting model, as opposed to interpolating from previously determined values.

directional light

A light source that shines from the same direction at every point in the scene. This is a handy shortcut for modeling far away light sources.

displacement

A distance in a particular direction. A vector exactly describes a displacement.

displacement vector

One of the vectors that defines a new coordinate system in terms of an old. The displacement vector is the vector from the old coordinate system's origin to the new coordinate system's origin.

display adapter

See "display controller."

display controller

A piece of computer hardware that receives drawing commands from the processor and drives the display. Some display controllers are commonly called the "video card", "display adapter", "graphics card", or something similar.

display controller, 2D

See "2D display controller."

display controller, 2 1/2 D

See "2 1/2 D display controller."

display controller, 3D

See "3D display controller."

display controller, GUI engine

See "2D display controller."

dither pattern

The particular pattern of threshold values used in dithering. Some dither patterns are random, while others are applied as repeating tiles across the whole image.

dithering

A technique for increasing an image’s apparent color (or gray scale) resolution without increasing the number of color (or gray) levels actually used, at the cost of adding a "grainy" look to the image.

dot pitch

A measure of how closely spaced the phosphor triads are on the face of a color CRT. The triads are arranged in a hexagonal pattern, and the dot pitch is the distance from the center of one triad to the center of any of its six neighbors. Dot pitch usually specified in millimeters. Typical values are from .2 to .3 mm.

dot product

A mathematical operation of two vectors that produces a scalar. If both vectors have a length of one (unit vectors), then the dot product is the perpendicular projection of one vector onto the other.

dot width

The width of a point, or dot, primitive. Unlike mathematical points, computer graphics point primitives must have finite width to be visible. Many graphics subsystems allow the application to specify a width for such point primitives.

DPI

Dots per inch. This is a common measure of how close individual dots are on some output devices, such as inkjet printers.

DRAM

Dynamic random access memory. Most computer main memories are implemented with DRAM.

drawing front end

See "front end."

dye sublimation printer

See "printer, dye sublimation."

- - - - E - - - -

electron gun

The part of a cathode ray tube (CRT) that emits the electron beam.

emissive color

An object surface property sometimes used with the Phong lighting model. An object’s emissive color is independent of any illumination. It therefore appears as if the object were emitting the color.

even field

See "field, even."

exclusive or

See "XOR operator."

explicit surface modeling

A class of modeling techniques that define objects by providing an explicit list of patches that cover their surfaces. These surface patches can be either polygons or any of a number of curved surface patch types. Other modeling techniques work on volumes, or only define an object’s surface implicitly.

eye point

The point, or coordinate, a scene is being viewed from. This is also called the "eye point" or "camera point".

eye ray

A ray in ray tracing that originated at the eye point. All recursively generated rays have an eye ray as their original ancestor.

eye vector

A vector from anywhere in the scene to the eye point. Eye vectors are usually unitized before use. The eye vector is needed in computing the apparent color when the object’s surface properties include specular reflection.

- - - - F - - - -

facet shading

A shading method where the shading normal vector is taken from the geometric normal of the surface actually drawn. This makes the surface patches visible, especially if they are planar.

field

Half of a complete video frame when interlacing is used. The two fields are referred to as the odd and the even. Each field contains only every other scan line.

field, even

The first of the two fields that make up a frame in interlaced video.

field, odd

The second of the two fields that make up a frame in interlaced video.

film recorder

An computer output device that can write images to film.

filter, anti-aliasing, box

See "anti-aliasing filter, box."

filter, anti-aliasing, good quality

See "anti-aliasing filter, good quality."

filter, box

See "anti-aliasing filter, box."

filter kernel

The function that defines the relative weight of a point depending on its position. The relative weight is used in computing a weighted average. This is actually a convolution operation, which is commonly used in anti-aliasing.

filtering

This is a broad word which can mean the removal of coffee grinds from the coffee. However, within the narrow usage of this book, a filtering operation is the same as a convolution operation (see "convolution"). Anti-aliasing is usually done by filtering.

flat projection

A method of projecting a 3D scene onto a 2D image such that the resulting object sizes are not dependent on their position. Flat projection can be useful when a constant scale is needed throughout an image, such as in some mechanical drawings.

flat shading

A shading method where each pixel of a primitive is drawn with the same color.

form factors

The name for the illumination coupling factors between polygons used in radiosity. Each form factor indicates how much light from one polygon will reach another polygon.

fractal

Something that has infinite detail. You will always see more detail as you magnify a small portion of a fractal. Mandelbrot set functions are examples of 2D fractals.

frame

One complete video image. When interlacing is used, a frame is composed of two fields, each containing only half the scan lines.

front end

The part of a display controller that receives drawing commands from the processor and writes the primitives into the bitmap.

- - - - G - - - -

gaze direction

The direction the virtual camera is pointed in the scene description. The center of the image will display whatever is along the gaze direction from the eye point.

geometric normal vector

A normal vector of the primitives as they are actually drawn. This often differs from the normal vector of the surface that was being modeled. Facet shading results when the shading normal vector is taken from the geometric normal vector.

GIF

A file format for storing images. GIF stands for Graphics Interchange format, and is owned by Compuserve, Inc.

Gouraud shading

See "interpolation, bi-linear."

graftal

A modeling technique where complex shapes are defined as relatively simple, recursive procedures.

graphic primitive

An object that the graphics system is capable of drawing into the bitmap. Examples are lines, points, and some polygons.

graphics card

See "display controller."

GUI engine

See "2D display controller."

- - - - H - - - -

hard copy

A copy of computer data that is directly readable by a human without using the computer. A printout is a good example of a hard copy.

HLS

Another name frequently used for the same thing as IHS. See "IHS."

homogeneous

Constant throughout.

horizontal refresh rate

See "refresh rate, horizontal."

HSV

Another name frequently used for the same thing as IHS. See "IHS."

HTML

HyperText Markup Language. The text formatting language used by documents on the world wide web.

- - - - I - - - -

IHS

A color space where colors are defined by their intensity, hue, and saturation attributes. This is sometimes referred to as HSV, which stands for "hue, saturation, value."

image

A two dimensional array of pixels that together form a picture.

image file

A computer file that contains an image.

implicit surface modeling

A class of modeling techniques that define an object by its surface, but without explicitly providing primitives that make up the surface. Examples are potential functions and iso-surfaces.

ink jet printer

See "printer, ink jet."

internet

The name for the global network connecting many computers to each other.

interpolation

The mathematical process of determining plausible in-between values, given explicit values at particular points. Pixel values of polygons are often interpolated from values explicitly calculated at the vertices. Interpolation is usually much faster than explicitly calculating values.

inverse dynamics

A method for specifying motion in an animation. Linkages and other constraints are defined for the objects. A final state is then specified for some of the objects, and the computer calculates the motion of all the objects so that the final state is reached. Unlike in inverse kinematics, dynamic properties are taken into account, such as momentum, friction, energy loss in collisions, etc.

inverse kinematics

A method for specifying motion in an animation. Linkages and other constraints are defined for the objects. A final state is then specified for some of the objects, and the computer calculates the motion of all the objects so that the final state is reached.

iso-surface

An implicit surface that exists wherever a continuous scalar field in a volume is at a particular value (the iso-value).

interpolation, bi-linear

See "bi-linear interpolation."

INTERSECTION operator

See "AND operator."

- - - - J - - - -

jaggies

See "aliasing."

JAVA

A platform-independent way of defining procedures for use with world wide web documents.

JPEG

A lossy image file compression technique for still images.

- - - - K - - - -

key frame

A selected frame of an animation at which all the scene state is defined. In the key frame animation method, the scene state at key frames is interpolated to create the scene state at the in-between frames.

key frame animation

An animation control method that works by specifying the complete scene state at selected, or key, frames. The scene state for the remaining frames is interpolated from the state at the key frames.

key frame interpolation, linear

- - - - L - - - -

laser printer

See "printer, laser."

left vector

A vector sometimes used to define the virtual camera orientation in a scene description. This is more typically done with an up vector.

light, ambient

See "ambient light."

light attenuation

The property of light to get dimmer with distance from the light source. Real light intensity is proportional to 1/R2, where R is the distance from the light source.

light, directional

See "directional light."

light, point

See "point light source."

light, point with 1/R2 falloff

See "point light source with 1/R2 falloff."

light ray

A ray in ray tracing that is launched from a point on an object towards a light source. The ray is used to determine whether the light source is illuminating the point. If the ray reaches the light source without hitting anything, then the light source is illuminating the point.

light source

A scene object that illuminates other objects.

light, spot

See "spot light source."

light vector

A vector from a point on an object towards a light source. Light vectors are usually unitized before use. A light vector is needed for each light source in computing the apparent color when the object’s surface properties include diffuse or specular reflection.

linear interpolation

Interpolation is the process of determining plausible in-between values, given explicit values at particular points. Linear means that the values fall along a line from one known point to the next. This means the value changes a fixed amount for a fixed-sized step. Sometimes the term "linear interpolation" is used to refer to "bi-linear interpolation".

linear shading

See "bi-linear interpolation."

logical operator

See "boolean operator."

lookat point

A point in the scene that will project to the center of the image. The gaze vector points from the eye point towards a lookat point.

lossless compression

A compression scheme (see "compression") where all data is preserved. The data may be compressed and de-compressed any number of times without causing any changes. The compression ratio of lossless compression schemes is generally lower than that of lossy schemes. Runlength and LZW encoding are examples of lossless compression schemes.

lossy compression

A compression scheme (see "compression") where some data may be irreversibly lost, in favor of a high compression ratio. Many lossy schemes can trade off between amount of loss and the compression ratio. JPEG and MPEG are examples of lossy compression schemes for images.

lookup table

See "color lookup table."

LUT

See "color lookup table."

LZW compression

A digital data compression scheme that works by identifying and compressing recurring patterns in the data. LZW stands for Lempel-Ziv and Welch. Unisys corporation now claims that LZW compression is covered by its U.S. patent number 4,558,302.

- - - - M - - - -

mach bands

An optical illusion caused by a sudden change in the rate of change of the brightness (discontinuities in the brightness’ second derivative). This can give the appearance of a light or dark line at the sudden change.

mandelbrot set

A popular mathematical function that exhibits fractal characteristics.

material properties

See "surface properties."

MINUS operator

A constructive solid geometry (CSG) modeling operation on two objects. For the operation "a MINUS b", the resulting object exists wherever A existed without being coincident with B. This operation can also be expresses "a AND (NOT b)."

modeling

As used in this book, the process of creating a description of an object or scene for the purpose of subsequent rendering.

modeling, explicit surface

See "explicit surface modeling."

modeling, implicit surface

See "implicit surface modeling"

modeling, level of detail

modeling, polygon

See "polygon modeling."

modeling, potential functions

See "potential function."

modeling, procedural

See "procedural modeling."

modeling, space subdivision

See "space subdivision modeling."

monitor

A piece of computer hardware for the live display of output. A monitor is different from a CRT, in that a monitor is the complete user-accessible unit that includes a case, power supply, knobs, etc. Many monitors use a CRT as the main imaging component, but other technologies are also used, such as flat panels.

MPEG

A lossy image file compression technique for motion pictures or animation sequences.

- - - - N - - - -

normal vector

A vector pointing straight our from (at right angles to) a surface.

normal vector, geometric

See "geometric normal vector."

NOT operator

A constructive solid geometry (CSG) modeling operation on one object. The resulting object exists only where the original object did not.

NURBS

Non Uniform Rational B-Splines. A particular type of surface spline for making curved surface patches in modeling complex shapes. These type of splines are supported by many computer aided design (CAD) systems.

- - - - O - - - -

object hierarchy ray tracing

See "ray tracing, object hierarchy."

octree

A hierarchical method for subdividing a volume. In this method, the volume is originally represented as one rectangular box (parallelpiped, to be more precise). If more detail is needed, the box is split exactly in half along each of its three major dimensions, yielding eight smaller boxes. This process is repeated on each sub-box until the desired level of detail is achieved, or an arbitrary subdivision limit is reached. Octrees are the 3D equivalent of quadtrees.

odd field

See "field, odd."

OpenGL

A 3D graphics procedural interface. It was developed by Silicon Graphics, Inc., based on their earlier proprietary GL graphics library.

OR operator

A constructive solid geometry (CSG) modeling operator on two objects. The resulting object exists where either or both input objects exist. This operation is also call UNION.

origin

The point in a coordinate space where all coordinates are zero.

orthographic projection

See "flat projection."

overlay where not zero

A simple image compositing method where the overlay image results wherever it’s not zero, and the background image results wherever the overlay image is zero. This method is rather simplistic, but is sometimes supported in low-end hardware. It can be useful in overlaying text, for example. Alpha buffering is a more general compositing method.

- - - - P - - - -

palette image format

A format for storing an image that works much like pseudo color in a display controller. Each pixel contains a color ID instead of the actual color value. The true color represented by each color ID is defined in a table called the palette, which must also be stored with the image. A palette is much like a LUT in a pseudo color display controller.

parametric animation

An animation control method where scene state is determined by mathematical functions or computer procedures that take animation time as an input parameter.

particle system

A modeling technique where objects are defined by the collective tracks of many individual particles. Randomness is usually used to determine the details for each particle automatically, although overall guidance is supplied by the user.

persistence of vision

The property of the human visual system to continue seeing an image a short time (fraction of a second) after it has gone away.

perspective projection

A method of projecting a 3D scene onto a 2D image such that distant objects are displayed smaller than near ones. A normal camera produces images using perspective projection.

phong lighting model

A particular method for computing the apparent color of an object at a particular point.

phong shading

A shading method where the shading normal vector is interpolated for each pixel, then used in a separate apparent color calculation for that pixel.

phosphor (or phosphors)

The material coating the inside of a CRT face. Phosphors have the special property that they emit light when struck by an electron beam.

phosphor persistence

The property of phosphors to stay lit a short time (fraction of a second) after the electron beam is cut off or moved away.

phosphor triad

One red, green, and blue phosphor dot on the face of a color CRT. The dots are arranged in an equilateral triangle. Triads are arranged in a hexagonal pattern, such that each triad is the same distance from each of its six neighbors.

pixel

The smallest indivisible unit of a digital image. A pixel is always the same color throughout. An image is a two dimensional array of pixels.

point light source

A light source where all the light comes from one point. Although real light gets dimmer farther from a light source, a computer graphics point light source shouldn’t be assumed to work that way unless explicitly stated.

point light source with 1/R2 falloff

A point light source where the light gets dimmer farther from the source. The light intensity is proportional to 1/R2, where R is the distance to the light source. This formula is used because that’s how real light works.

point primitive

A graphic primitive that looks like a dot, or point.

polygon primitive

A graphic primitive that is an area enclosed by a loop of straight edges. Triangles and squares are examples of polygon primitives.

polygon modeling

A modeling method where surfaces are approximated with abutting polygons.

polyline primitive

A graphic primitive of end-to-end line segments. A polyline primitive is more efficient than each of the line segments as separate vector primitives.

potential function

A type of implicit surface. See the text for details.

primitive

See "graphics primitive."

primitive, point

See "point primitive."

primitive, polygon

See "polygon primitive."

primitive, polyline

See "polyline primitive."

primitive, quad mesh

See "quad mesh primitive."

primitive, vector

See "vector primitive."

printer

Computer peripherals for producing hard copy on paper and other similar media.

printer, dye sublimation

A type of printer that works by evaporating controlled amounts of dye from a ribbon onto the page. The amount of dye can be accurately controlled, yielding continuous color resolution. Dye sublimation printers are relatively expensive, but produce output comparable in quality to the traditional wet silver photographic process.

printer, ink jet

A type of printer that shoots tiny droplets of ink onto the page. Ink jet printers are a relatively low cost way to get computer graphics output.

printer, laser

A type of printer that works almost like a photocopier. The image is created by a laser under computer control, instead of coming from an original document as in a photocopier.

printer, thermal wax

See "printer, wax transfer."

printer, wax transfer

A type of printer that works by depositing small specs of wax from a ribbon onto the page. The ribbon is pressed against the page, and wax is transferred wherever the ribbon is heated. This type of printer is also called "thermal wax."

procedural model

An object model defined implicitly by a procedure that can produce volume or surface elements.

projection, flat

See "flat projection."

projection method

A scheme for mapping the 3D scene geometry onto the 2D image.

projection, orthographic

See "flat projection."

projection, perspective

See "perspective projection."

pseudo color system

A graphics system that stores pseudo colors, instead of true colors, in its bitmap. Pseudo colors are translated to true colors by the color lookup table (LUT).

pseudo color value

See "color index value."

- - - - Q - - - -

quad mesh primitive

A graphic primitive that contains a grid of quadrilaterals. A quad mesh primitive is more efficient than the equivalent separate quadrilateral primitives.

quadtree

A hierarchical method for subdividing an area. In this method, the area is originally represented as one box (parallelogram, to be precise). If more detail is needed, the box is split exactly in half along each of its two major dimensions, yielding four smaller boxes. This process is repeated on each sub-box until the desired level of detail is achieved, or an arbitrary subdivision limit is reached. Quadtrees are the 2D equivalent of octrees.

- - - - R - - - -

radiosity

A rendering method that takes into account diffuse reflection between objects.

raster scan

The name for the pattern the electron beam sweeps out on a CRT face. The image is made of closely spaced scan lines, or horizontal sweeps.

ray casting

A term sometimes used to denote non-recursive ray tracing.

ray, eye

See "eye ray."

ray, light

See "light ray."

ray tracing

A rendering method that follows rays of light backwards to eventually find what color they are. Rays may be launched recursively when the color of a point on an object depends on incoming light from specific known directions. For example, when a ray hits a reflective object, a recursive ray is launched in the direction the reflected light is coming from.

ray tracing, object hierarchy

A ray tracing speedup method where objects are kept track of in groups. These groups can be further grouped in a hierarchy. A bounding volume is maintained for each group in the hierarchy. If a ray doesn't intersect a bounding volume, then it definitely doesn't intersect any subordinate object in the hierarchy.

ray tracing, space subdivision

A ray tracing speedup method where the scene space is subdivided into blocks. A list is kept for each block indicating which objects a ray could hit in that block. As a ray is traced, it is walked thru the blocks, checking for intersection only with the objects listed in each block.

ray tracing speed issues

reasoning, circular

See "circular reasoning."

recursive ray tracing

See "ray tracing."

reflection vector

A vector used in the phong lighting model to compute the specular reflection. It is usually unitized, and points in the direction light is reflecting off the object.

refresh rate

The rate at which parts of the image on a CRT are re-painted, or refreshed. The horizontal refresh rate is the rate at which individual scan lines are drawn. The vertical refresh rate is the rate at which fields are drawn in interlaced mode, or whole frames are drawn in non-interlaced mode.

refresh rate, horizontal

The rate at which scan lines are drawn when the image on a CRT is re-drawn, or refreshed.

refresh rate, vertical

The rate at which fields are re-drawn on a CRT when in interlaced mode, or the rate at which the whole image is re-drawn when in non-interlaced mode.

regular BSP tree

See "BSP tree, regular."

rendering

The process of deriving an 2D image from the 3D scene description. This is basically the "drawing" step.

resolution

The measure of how closely spaced the pixels are in a displayed image. For example, if 1,024 pixels are displayed across a screen that is 12 inches wide, then the image has a resolution of 85 pixels per inch.

RGB

A color space where colors are defined as mixtures of the three additive primary colors red, green, and blue.

right vector

A vector sometimes used to define the virtual camera orientation in a scene description. This is more typically done with an up vector.

runlength encoding

A lossless digital data compression scheme. It identifies identical consecutive values, and replaces them with just one copy of the value and a repeat count.

- - - - S - - - -

scalar

A regular, single number, as opposed to a vector.

scan line

One line in a raster scan. Also used to mean one horizontal row of pixels.

scan line order

A way of arranging image data so that all the pixels for one scan line are stored or transmitted before the pixels for the next scan line.

scan rate

See "refresh rate."

scattered light

See "secondary scatter."

scene

The complete 3D description of everything needed to render an image. This includes all the object models, the light sources, and the viewing geometry.

secondary scatter

Light that is reflected from a non-emitting object that illuminates other objects. This is the kind of inter- object illumination that is computed by radiosity.

shading

This term is used several ways in computer graphics. However, shading always has something to do with figuring out pixel color values, as opposed to figuring out the geometry or which pixels are to be drawn.

shading, bi-linear

See "bi-linear interpolation."

shading, facet

See "facet shading."

shading, flat

See "flat shading."

shading, Gouraud

See "bi-linear interpolation."

shading, linear

See "bi-linear interpolation."

shading normal vector

The normal vector used in determining the apparent color of an object. In facet shading, the shading normal is the geometric normal of the primitives used to model the surface. In smooth shading, the shading normal is the normal vector of the surface that was modeled. The shading normal vector may be further modified by bump mapping.

shadow mask

A thin layer in a color CRT that the electron beams can not penetrate. It is suspended just in front of (from the electron beam’s point of view) the phosphor screen. The shadow mask has one hole for each phosphor color triad. Due to the position of the phosphor triads, the shadow mask holes, and the angle of the three electron beams, each electron beam can only hit the phosphor dots of its assigned color.

SIGGRAPH

The Special Interest Group on Graphics of the Association for Computing Machinery (ACM). SIGGRAPH is the premier professional association for computer graphics.

smooth shading

Any shading technique that attempts to make the rendered surface look as close as possible to what was modeled, instead of the primitives used to approximate that model. This is usually done by taking the shading normal vector from modeled surface, instead of from the geometric normal vector.

solid texture

See "3D texture."

space subdivision, adaptive

See "adaptive space subdivision."

space subdivision modeling

A modeling technique where the scene is broken into small regions of space. A list of the objects present is kept for each region. Examples of space subdivision modeling are BSP trees and octrees.

space subdivision ray tracing

See "ray tracing, space subdivision."

specular color

The color of an object’s shiny highlights. Specular reflection also depends on the specular exponent. Both these parameters are part of the object’s surface properties.

specular exponent

This is a scalar value that controls the "tightness" of the specular highlights. A value of zero causes the specular color to be reflected equally in all direction, just like the diffuse color. An infinite value causes it to be reflected the same way a mirror would. Common values are about 5 to 60.

spline, B

See "B spline."

spline, beta

See "beta spline."

spline patch

A curved surface which takes its shape from the placement of a set of control points.

spot light source

A light source that does not shine in all directions, usually in a cone. This is a convenient hack for modeling lamps with shades or reflectors, without actually having to compute the effect of the shade or reflector during rendering.

subpixel

An input pixel to an operation that uses multiple smaller pixels at higher resolution to compute each resulting pixel at the final resolution. This is done, for example, in an anti-aliasing filtering operation.

surface orientation, apparent

See "apparent surface orientation."

surface properties

The collective name for all the object-specific parameters that are used to compute the apparent color of a point on that object. These include the diffuse color, specular color, etc. The term "surface properties" is common but not standard. Other names for the same thing are "visual properties", or "material properties".

- - - - T - - - -

temporal aliasing

Aliasing in time by animation frame, instead of spatially by pixel. The visual effect of temporal aliasing has been called "strobing". Moving objects appear to jump thru a sequence of frozen steps, instead of in smooth paths.

tessellation level

The relative fineness or granularity of the patches used to model a surface. A model with smaller, and therefore more patches is said to have a higher tessellation level.

tessellation

The act of tiling a surface with individual surface patches.

texil

One pixel of a texture map, where the texture map is an image. In diffuse color texture mapping, the texture value is a color. The texture color values are therefore often supplied as an image. A texil is one pixel of this image, as opposed to one pixel of the final image being rendered.

texture, 3D

See "3D texture."

texture map

The external function that supplies the texture value in texture mapping. In diffuse color texture mapping, the texture map is a set of color values in two dimensions. This is usually specified as an image. Note, however, that not all texture maps are images, because not all textures are two dimensional color values. Bump mapping is a form of texture mapping where the texture is not an image, since the texture values are shading normal vector perturbations instead of colors.

texture mapping

The process where some parameter of the apparent color computation is replaced by an external function, called the texture map. A common example is texture mapping an object’s diffuse color from an image. This gives the appearance of the image painted onto the object. Bump mapping is another form of texture mapping discussed in this book.

texture mapping, diffuse color

A form of texture mapping where the texture defines the object’s diffuse color. This gives the appearance of pasting the texture image onto the object.

texture mapping, normal vector perturbations

See "bump mapping."

thermal wax printer

See "printer, wax transfer."

TIFF

Tag Image File Format. A common image file format. Just about all applications that can import image files support the TIFF format.

toner

The material that is deposited onto the page to form the image in a photocopier or laser printer.

transform, 3D

See "3D transform."

transformation matrix, 3D

See "3D transform."

transparency value

See "alpha value."

triad

See "phosphor triad."

triangle strip primitive

A graphic primitive that contains a set of successively abutting triangles. A triangle strip primitive is more efficient that the equivalent separate triangle primitives. It is also called a "Tstrip."

true color system

A graphics system that stores true colors, as opposed to pseudo colors, in its bitmap.

Tstrip primitive

See "triangle strip primitive."

- - - - U - - - -

UNION operator

See "OR operator."

unit vector

A vector of length, or magnitude, one.

unitize

To make a vector a unit vector. This means adjusting its length to be one without effecting the direction it’s pointing.

up vector

A vector used to define the orientation of the virtual camera in a scene description.

- - - - V - - - -

vector

A value that has a direction and a magnitude (length). Plain numbers can be called "scalars" to explicitly distinguish them from vectors.

vector addition

The process of adding two vectors, which results in another vector.

vector, basis

See "basis vector."

vector cross product

See "cross product."

vector dot product

See "dot product."

vector, eye

See "eye vector."

vector, geometric normal

See "geometric normal vector."

vector, left

See "left vector."

vector, light

See "light vector."

vector primitive

A graphics primitive that looks like a line segment. Unlike mathematical line segments, vector primitives have finite widths.

vector, reflection

See "reflection vector."

vector, right

See "right vector."

vector scaling

An operation of a vector and a scalar. The resulting vector is the same as the input vector, except that its magnitude (length) is multiplied by the scalar. Scaling a vector by the reciprocal of its magnitude results in a unit vector.

vector, shading normal

See "shading normal vector."

vector thickness

The width of a vector, or line segment, primitive. Unlike mathematical line segments, computer graphics vector primitives must have a finite width to be visible. Many graphics systems allow the application to specify a width for such primitives.

vector, unit

See "unit vector."

vector unitize

See "unitize."

vector, up

See "up vector."

vertical refresh rate

See "refresh rate, vertical."

video back end

The part of a display controller that reads the pixel data in the bitmap and produces the live video signals. Among other components, the back end contains the color lookup tables (LUTs), the digital to analog converters (DACs), and the logic to generate the video timing signals.

video card

See "display controller."

view direction

See "gaze direction."

view point

See "eye point."

virtual reality

A name loosely applied to systems that attempt to immerse the user in a virtual world generated by the computer. This is often done with a stereoscopic display that is updated based on head position, and usually includes some sort of 3D pointing device. More advanced systems include 3D sound and some form of touch feedback. This is still a rapidly changing area.

visual properties

See "surface properties."

voxel

A unit of subdivided space. Voxel stands for "volume element". Nodes in an octree, for example, are referred to as voxels.

VRAM

Video random access memory. This is really DRAM with additional features specifically for use as bitmap memory in display controllers. VRAM typically costs twice as much as DRAM, but allows the drawing engine full access to the bitmap independent from the video back end. This can increase the hardware drawing rate.

VRML

Virtual Reality Modeling Language. The standard description language for 3D graphics in world wide web documents.

- - - - W - - - -

wax transfer printer

See "printer, wax transfer."

web browser

Software that can read and display world wide web documents.

wire frame

A rendering method where only the outline of objects and primitives are drawn.

world wide web

A set of interconnected documents on the internet that adhere to the HTML standard. The world wide web is often abbreviated to WWW.

WWW

See "world wide web."

- - - - X - - - -

XOR operator

A constructive solid geometry (CSG) modeling operation on two objects. The resulting object exists where each object existed alone, not coincident with each other. XOR is an abbreviation for "exclusive or."

X Windows

A window management and 2D graphics environment that is very common on Unix systems.

- - - - Y - - - -

- - - - Z - - - -

Z buffer

The collective name for all the Z values in a bitmap.

Z buffer rendering

A rendering method where an additional depth, or Z, value is kept per pixel. A pixel is only overwritten by a new primitive if the new Z value represents a shorter distance, or depth, from the eye point. The end result is an image of the front most surface of the front most objects.

Z value

The name for the value that represents distance from the eye point in a Z buffer rendering system.

Geometry for Computer Graphics

Mike Bailey

San Diego Supercomputer Center
University of California San Diego

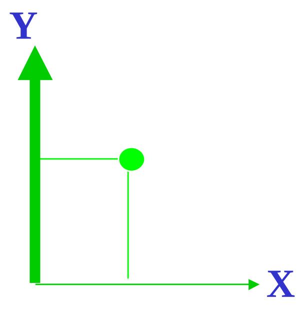
`mjb@sdsc.edu`

`http://www.sdsc.edu/~mjb`

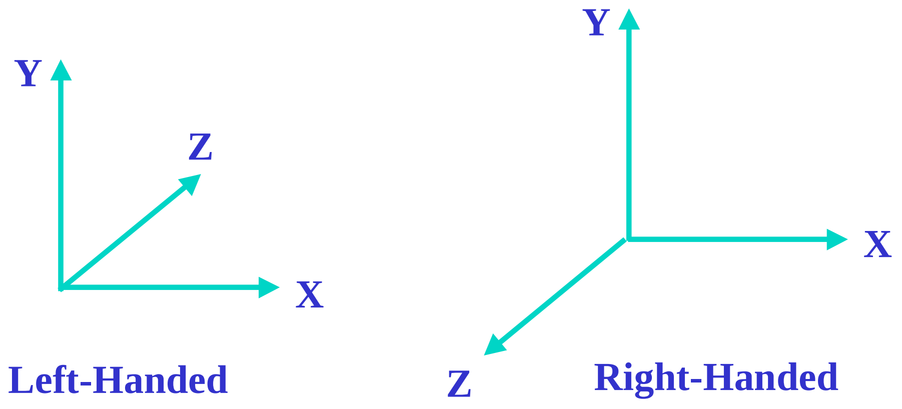
Geometry is Our Friend

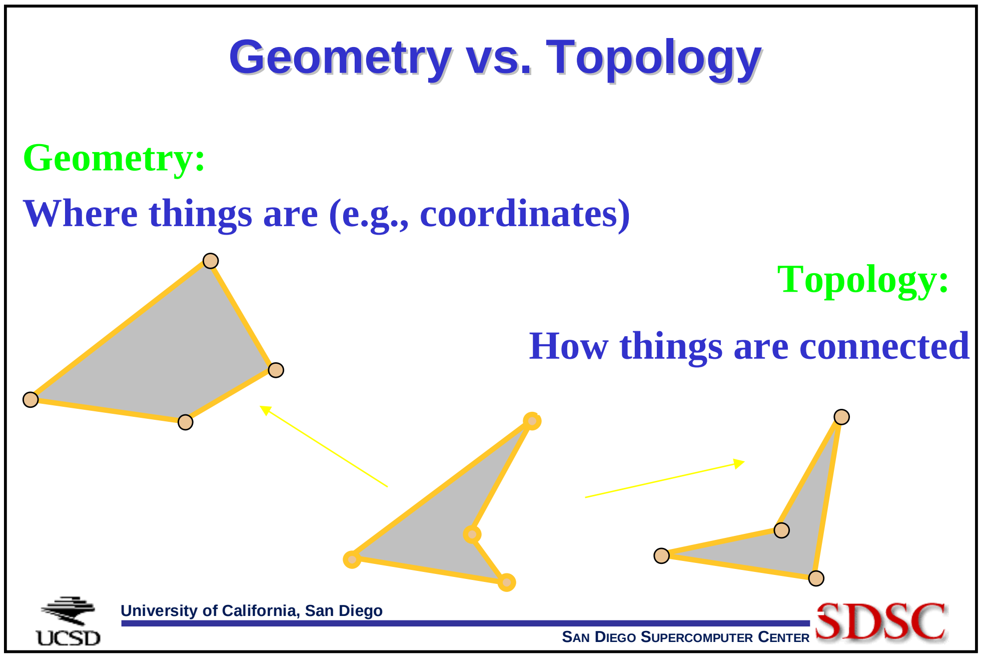
(no, really · · ·)

2D Coordinate System



3D Coordinate Systems





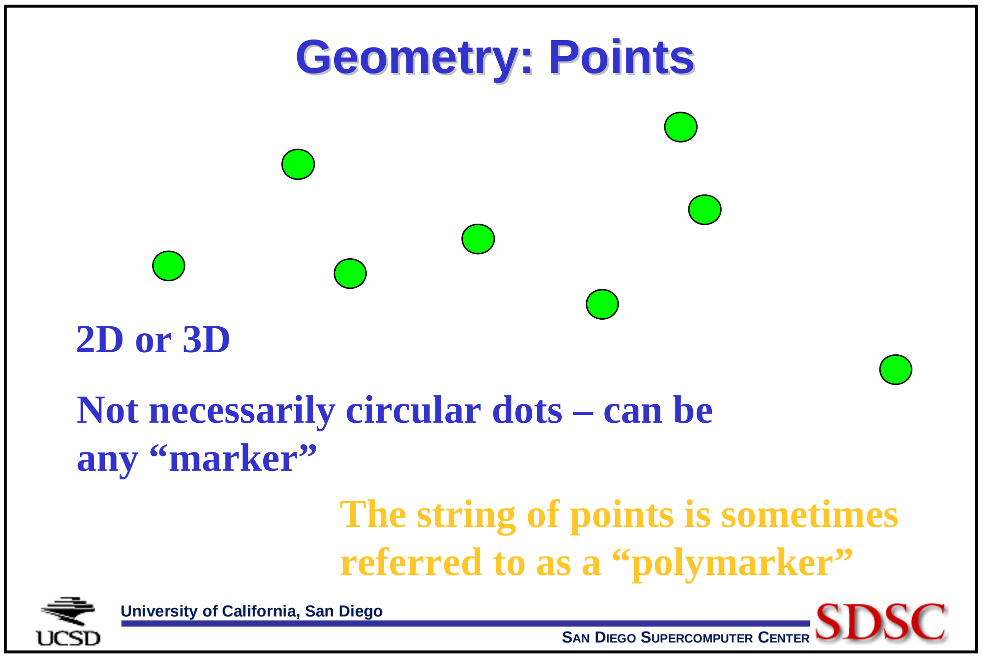
Geometry vs. Topology

Geometry:

Where things are (e.g., coordinates)

Topology:

How things are connected



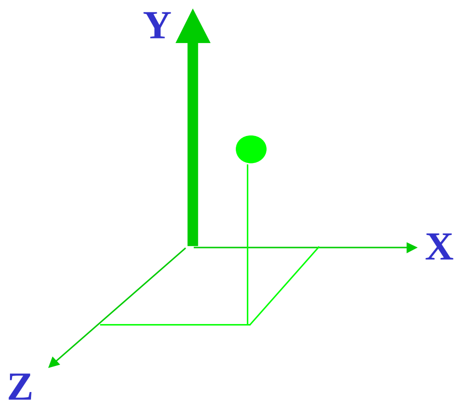
Geometry: Points

2D or 3D

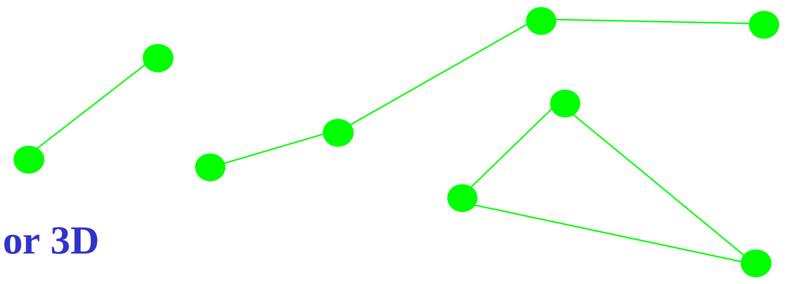
Not necessarily circular dots – can be any "marker"

The string of points is sometimes referred to as a "polymarker"

3D Points



Lines



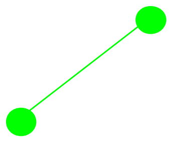
2D or 3D

Defined as a list of points

Topology varies

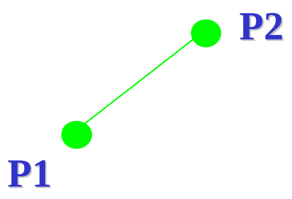
Sometimes referred to as a “polyline”

Why “y=mx+b” Isn't Good for Graphics Applications Software



- Cannot represent vertical lines (m = inf)
- Can only represent infinite lines, not finite line segments
- Can only represent 2D lines, not 3D

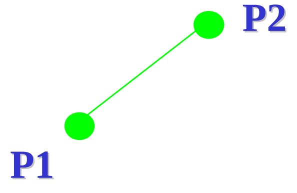
Parametric Line Equation



```
x = X1 + t * ( X2 - X1 )
y = Y1 + t * ( Y2 - Y1 )
x = Z1 + t * ( Z2 - Z1 )

                    0.0 <= t <= 1.0
```

Can Also Be Thought of As a Blending Function



```
x = (1-t) * X1  +  t * X2
y = (1-t) * Y1  +  t * Y2
z = (1-t) * Z1  +  t * Z2
```

```
0.0 <= t <= 1.0
```

Linear Blending Shows Up in a Lot of Computer Graphics Applications

You can linearly blend any two quantities with:

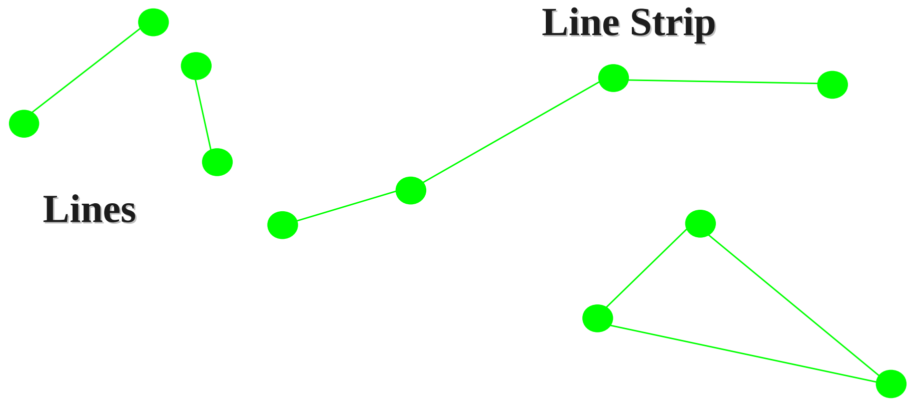
```
q = Q1 + t * ( Q2 - Q1 )
```

or, if you'd prefer:

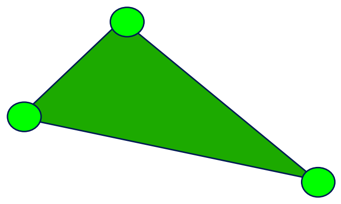
```
q = (1 - t ) * Q1  +  t * Q2
```

color, shape, location, angle, scale factors, •••

Line Topologies

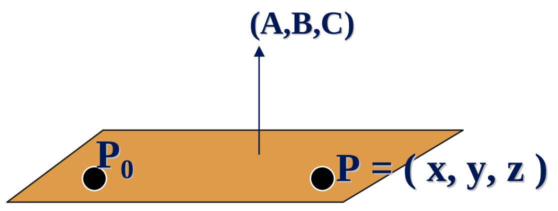


Polygons



- Planar
- Defined as a closed sequence of points
- 2D or 3D

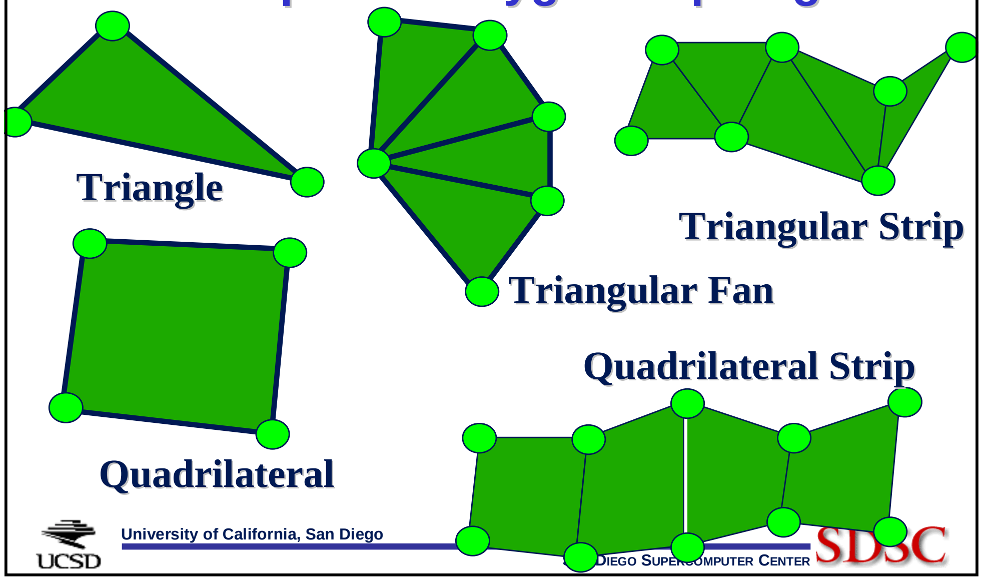
Sidebar: What is "Planar?"



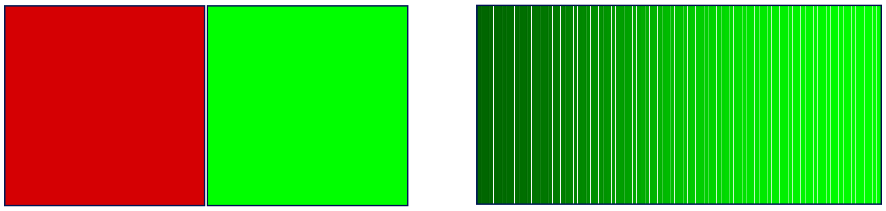
If the point P is above / on / below the plane, then:

$$Ax + By + Cz - (Ax_0 + By_0 + Cz_0) \begin{matrix} > \\ = \\ < \end{matrix} 0$$

Some Special Polygon Topologies

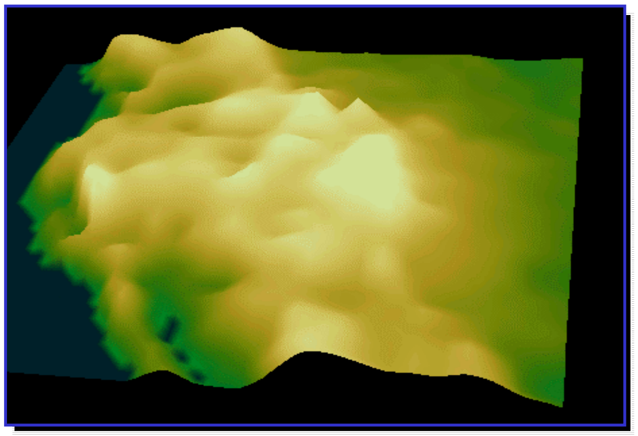


Polygon Patterns: Color Interpolation



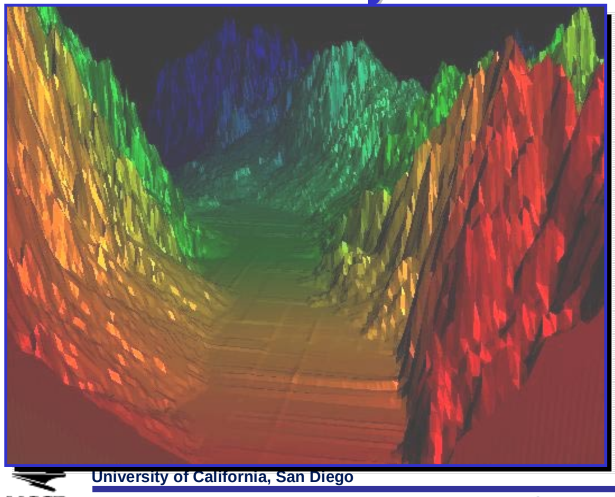
Referred to as *Smooth Shading*, or *Gouraud Shading*

Polygon Patterns: Color Interpolation

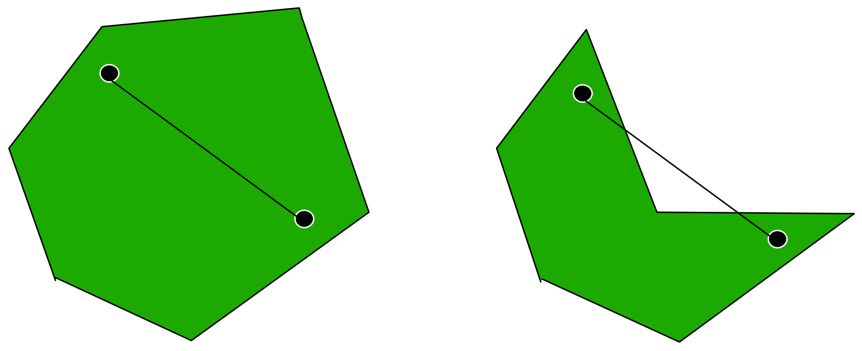


Polygon Patterns: Texture Mapping

Texture Mapping: Automatically-Generated Textures



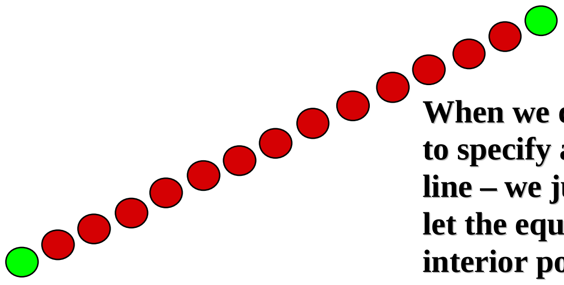
Convex vs. Concave



Convex

Concave

Higher Order Geometry

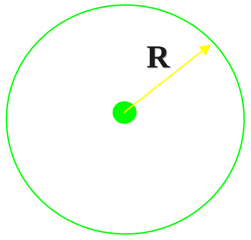


When we draw a line, we do not need to specify all pixel points along the line – we just give the endpoints and let the equation determine the interior points

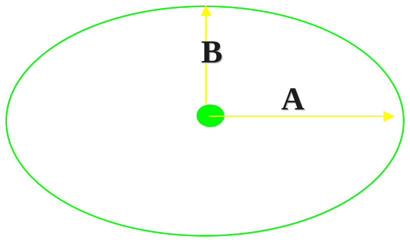
Can we do the same with other curve and surface types?

Conics

Circle



Ellipse



x = Rcosθ

y = Rsinθ

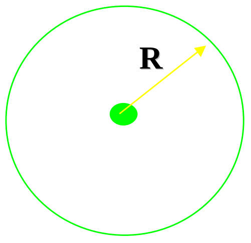
x = Acosθ

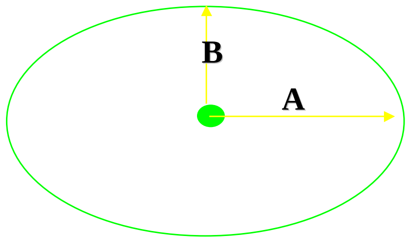
y = Bsinθ

Parabola, Hyperbola, • • •

0.0 ≤ t ≤ 1.0

It is often handy to think of the independent parameter as consistently varying from 0.0 to 1.0





x = Rcos(360t)

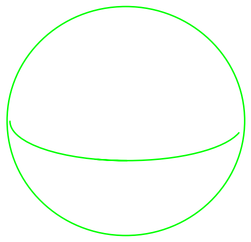
y = Rsin(360t)

x = Acos(360t)

y = Bsin(360t)

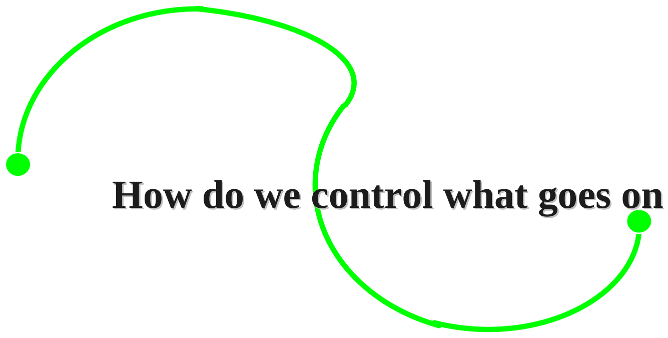
Quadrics

Sphere



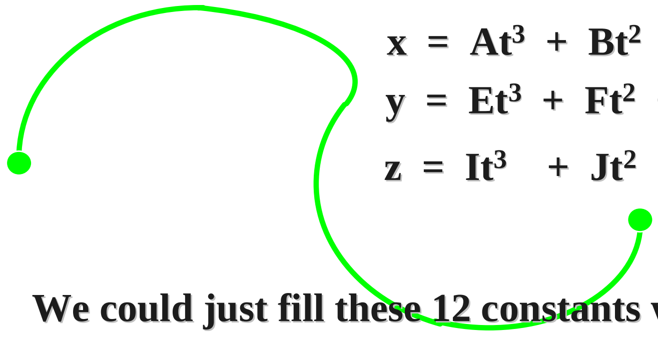
Ellipsoid, Paraboloid, Hyperboloid , [Torus,] • • •

Arbitrary Curves



How do we control what goes on in here?

Cubic Curves

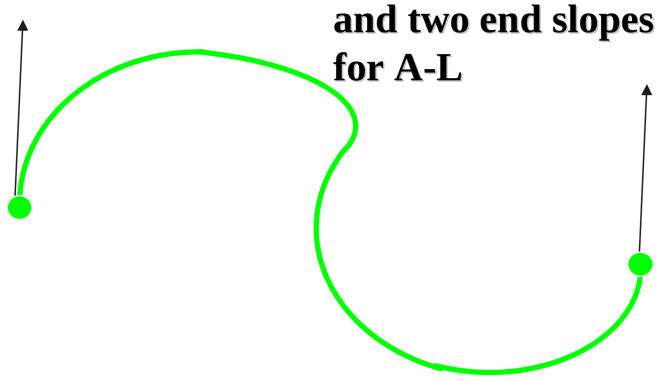


$$x = At^3 + Bt^2 + Ct + D$$
$$y = Et^3 + Ft^2 + Gt + H$$
$$z = It^3 + Jt^2 + Kt + L$$
$$0.0 \le t \le 1.0$$

We could just fill these 12 constants with random numbers, but there must be a better way

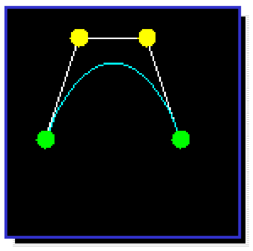
Hermite, or Coons, Cubic Curve

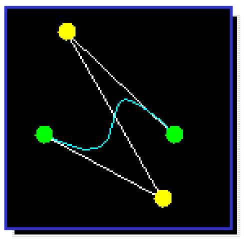
Specify the two end points and two end slopes – solve for A-L



Bézier Cubic Curves

Specify the two end points and two "control points" – solve for A-L



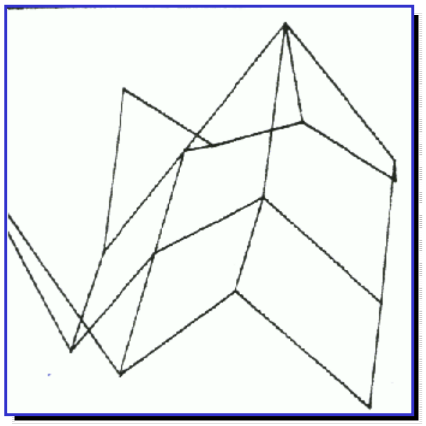


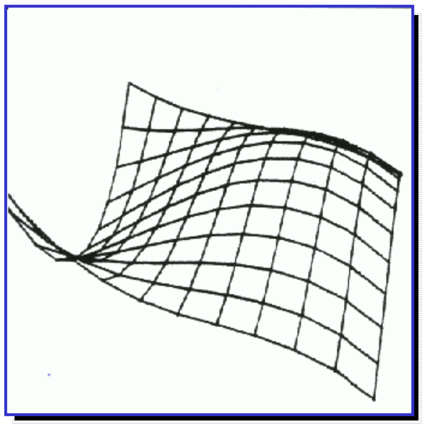
BiCubic Surfaces

$$X(s,t) = \begin{array}{l} As^3t^3 + Bs^3t^2 + Cs^3t + Ds^3 + \\ Es^2t^3 + Fs^2t^2 + Gs^2t + Hs^2 + \\ Ist^3 + Jst^2 + Kst + Ls + \\ Mt^3 + Nt^2 + Ot + P \end{array}$$

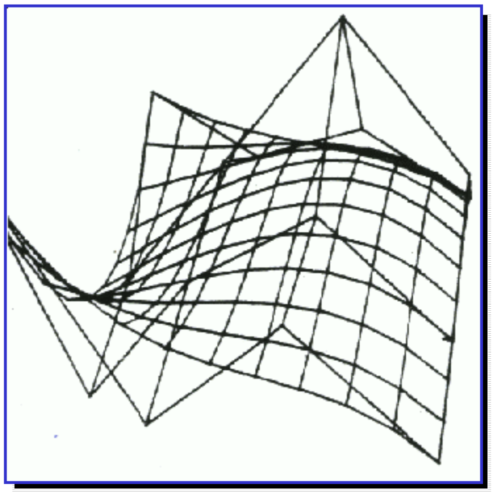
$$0.0 \leq s,t \leq 1.0$$

Bézier Bicubic Surfaces





Bézier Bicubic Surfaces



Transformations

Mathematical equations to change an object's coordinates

Linear Equations

```
x' = Ax + By + Cz + D

y' = Ex + Fy + Gz + H

z' = Ix + Jy + Kz + L
```

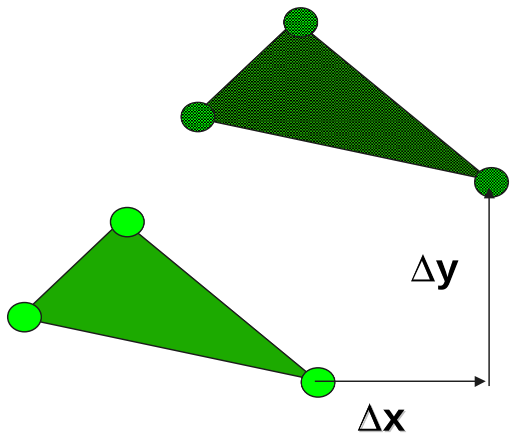
Transform the geometry – leave the topology as is

Translation

$x' = x + \Delta x$

$y' = y + \Delta y$

$z' = z + \Delta z$

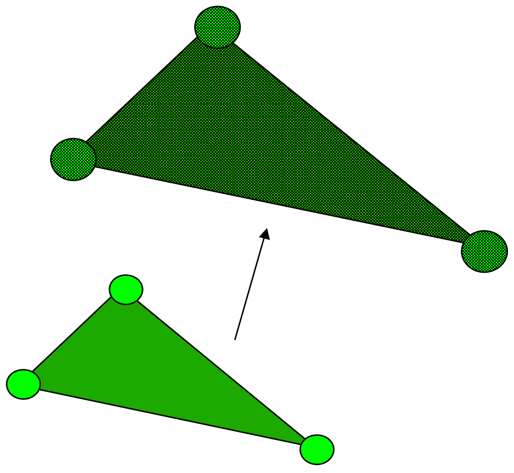


Scaling

$x' = x * Sx$

$y' = y * Sy$

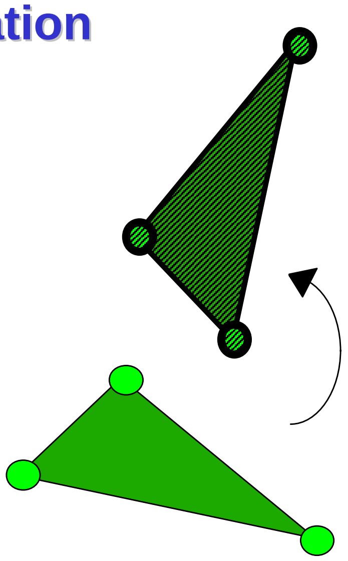
$z' = z * Sz$



2D Rotation

$x' = x\cos\theta - y\sin\theta$

$y' = x\sin\theta + y\cos\theta$



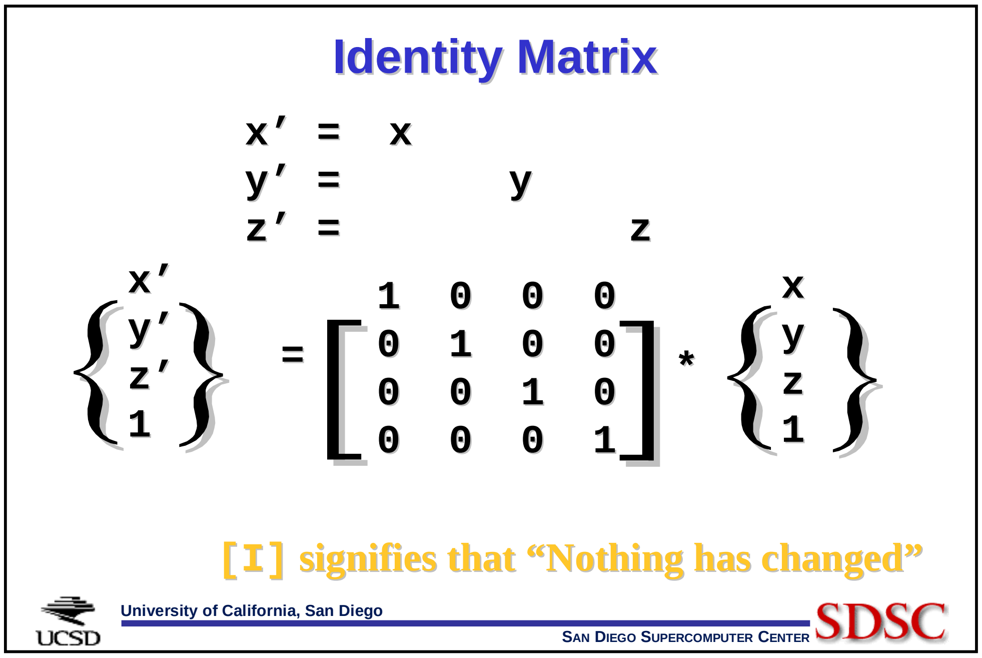
University of California, San Diego
SAN DIEGO SUPERCOMPUTER CENTER

Linear Equations in Matrix Form

$$x' = Ax + By + Cz + D$$
$$y' = Ex + Fy + Gz + H$$
$$z' = Ix + Jy + Kz + L$$

$$\begin{Bmatrix} x' \\ y' \\ z' \\ 1 \end{Bmatrix} = \begin{bmatrix} A & B & C & D \\ E & F & G & H \\ I & J & K & L \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix}$$

University of California, San Diego
SAN DIEGO SUPERCOMPUTER CENTER



Identity Matrix

```
x' = x
y' =      y
z' =           z
```

$$\begin{Bmatrix} x' \\ y' \\ z' \\ 1 \end{Bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix}$$

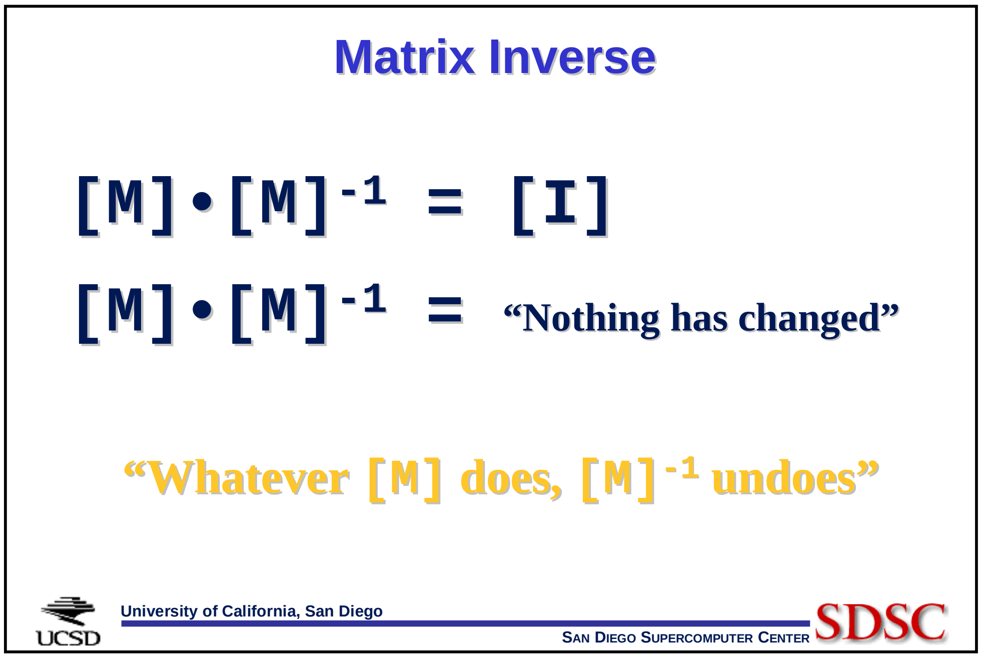
[I] signifies that “Nothing has changed”

University of California, San Diego

UCSD

San Diego Supercomputer Center

SDSC



Matrix Inverse

$[M] \bullet [M]^{-1} = [I]$

$[M] \bullet [M]^{-1} =$ “Nothing has changed”

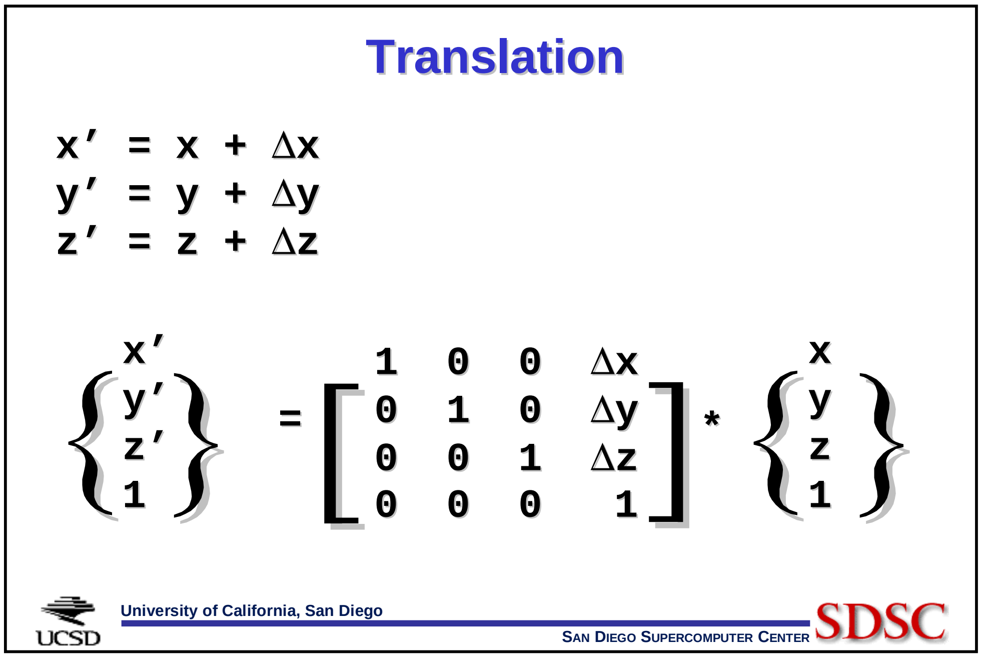
“Whatever [M] does, $[M]^{-1}$ undoes”

University of California, San Diego

UCSD

San Diego Supercomputer Center

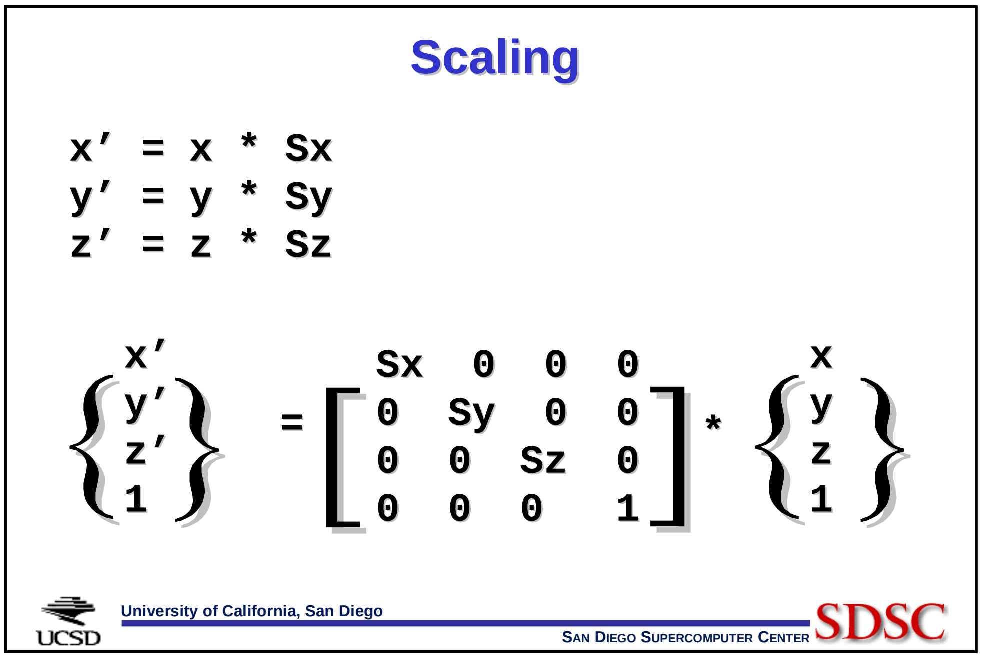
SDSC



Translation

$$x' = x + \Delta x$$
$$y' = y + \Delta y$$
$$z' = z + \Delta z$$

$$\begin{Bmatrix} x' \\ y' \\ z' \\ 1 \end{Bmatrix} = \begin{bmatrix} 1 & 0 & 0 & \Delta x \\ 0 & 1 & 0 & \Delta y \\ 0 & 0 & 1 & \Delta z \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix}$$



Scaling

$$x' = x * Sx$$
$$y' = y * Sy$$
$$z' = z * Sz$$

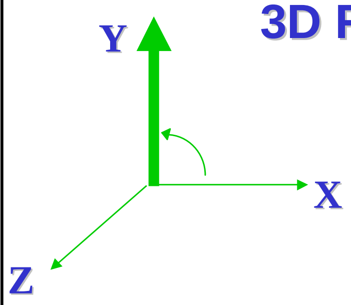
$$\begin{Bmatrix} x' \\ y' \\ z' \\ 1 \end{Bmatrix} = \begin{bmatrix} Sx & 0 & 0 & 0 \\ 0 & Sy & 0 & 0 \\ 0 & 0 & Sz & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix}$$

2D Rotation

```
x' = xcosθ - ysinθ
y' = xsinθ + ycosθ
```

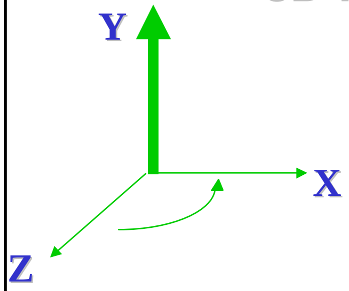
$$\begin{Bmatrix} x' \\ y' \\ z' \\ 1 \end{Bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix}$$

3D Rotation About Z



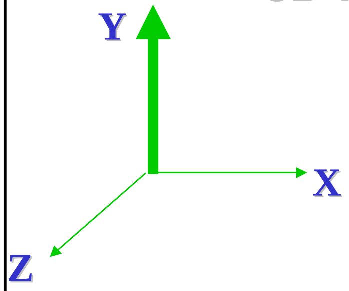
$$\begin{Bmatrix} x' \\ y' \\ z' \\ 1 \end{Bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix}$$

3D Rotation About Y



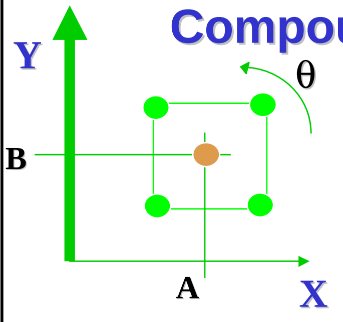
$$\begin{Bmatrix} x' \\ y' \\ z' \\ 1 \end{Bmatrix} = \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix}$$

3D Rotation About X



$$\begin{Bmatrix} x' \\ y' \\ z' \\ 1 \end{Bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix}$$

Compound Transformations



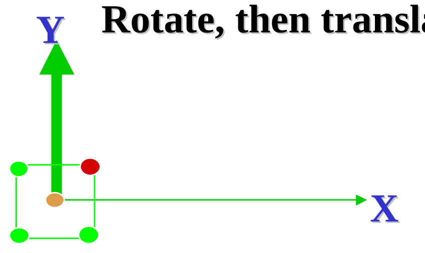
Write it $\longrightarrow$

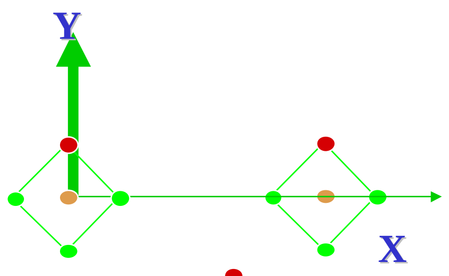
$$\begin{matrix}x'\\y'\\z'\\1\end{matrix} = \left([T_{+A,+B}] * \left([R_\theta] * \left([T_{-A,-B}] * \begin{matrix}x\\y\\z\\1\end{matrix}\right)\right)\right)$$

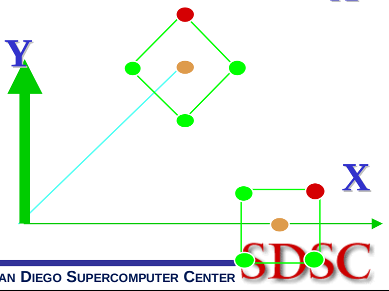
$\longleftarrow$ Say it

Matrix Multiplication is Not Commutative

Rotate, then translate







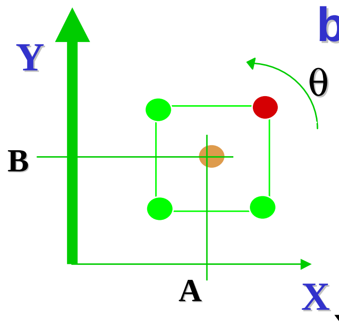
Translate, then rotate

Matrix Multiplication is Associative

$$\begin{matrix} x' \\ y' \\ z' \\ 1 \end{matrix} = \left([T_{+A,+B}] * \left([R_{\theta}] * \left([T_{-A,-B}] * \begin{matrix} x \\ y \\ z \\ 1 \end{matrix} \right) \right) \right)$$

$$\begin{matrix} x' \\ y' \\ z' \\ 1 \end{matrix} = \left(\left([T_{+A,+B}] * \left([R_{\theta}] * [T_{-A,-B}] \right) \right) * \begin{matrix} x \\ y \\ z \\ 1 \end{matrix} \right)$$

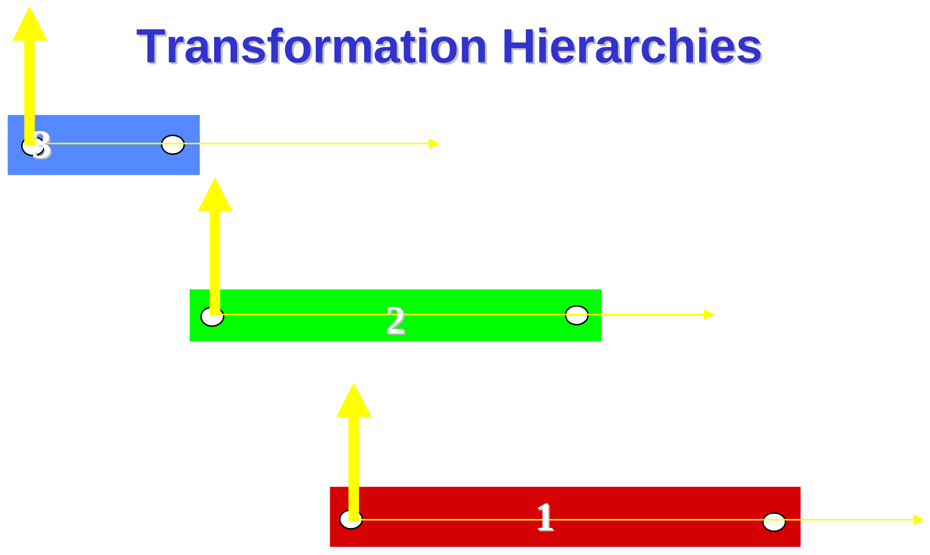
Can Multiply All Geometry by One Matrix !



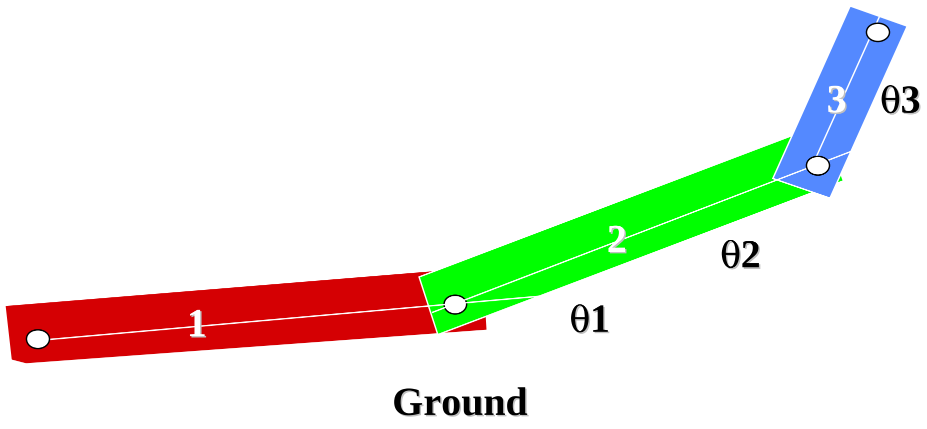
$$\begin{matrix} x' \\ y' \\ z' \\ 1 \end{matrix} = \left(\left(\mathbf{M} \right) * \begin{matrix} x \\ y \\ z \\ 1 \end{matrix} \right)$$

Hardware can do this very quickly!

Transformation Hierarchies



Transformation Hierarchies



Positioning Part #1 With Respect to Ground

1. Rotate by θ1
2. Translate by $\Delta_{1/G}$

Write it $\longrightarrow$

$$[M_{3/G}] = [T_{1/G}] * [R_{\theta 1}]$$
$$= [M_{1/G}]$$

Say it $\longleftarrow$

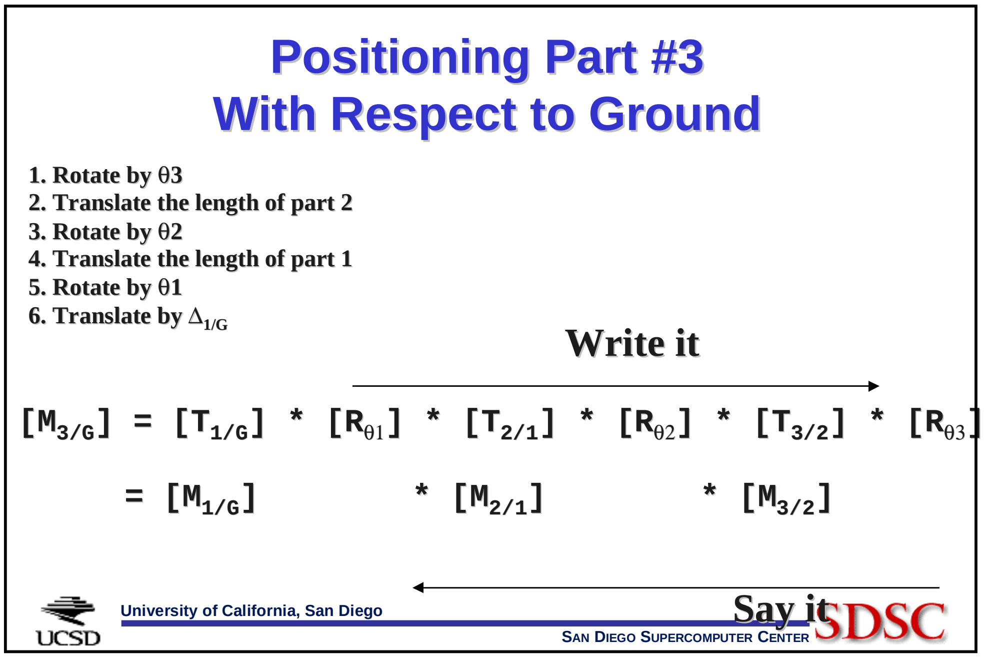
Positioning Part #2 With Respect to Ground

1. Rotate by θ2
2. Translate the length of part 1
3. Rotate by θ1
4. Translate by $\Delta_{1/G}$

Write it $\longrightarrow$

$$[M_{2/G}] = [T_{1/G}] * [R_{\theta 1}] * [T_{2/1}] * [R_{\theta 2}]$$
$$= [M_{1/G}] \quad * [M_{2/1}]$$

Say it $\longleftarrow$



Positioning Part #3 With Respect to Ground

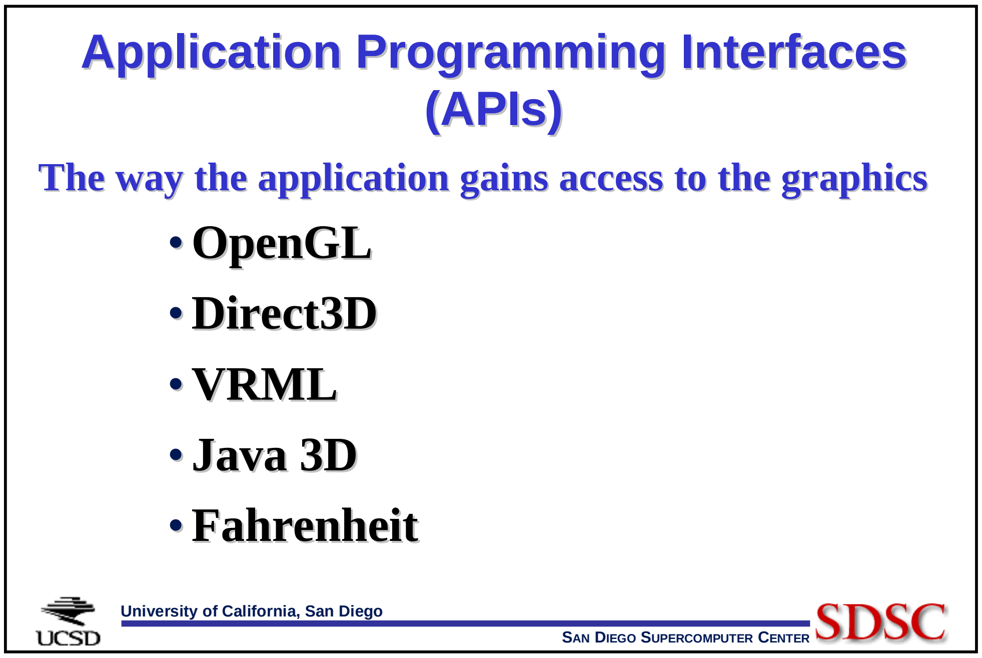
1. Rotate by θ3
2. Translate the length of part 2
3. Rotate by θ2
4. Translate the length of part 1
5. Rotate by θ1
6. Translate by $\Delta_{1/G}$

Write it $\longrightarrow$

$$[M_{3/G}] = [T_{1/G}] * [R_{\theta 1}] * [T_{2/1}] * [R_{\theta 2}] * [T_{3/2}] * [R_{\theta 3}]$$

$$= [M_{1/G}] * [M_{2/1}] * [M_{3/2}]$$

$\longleftarrow$ Say it

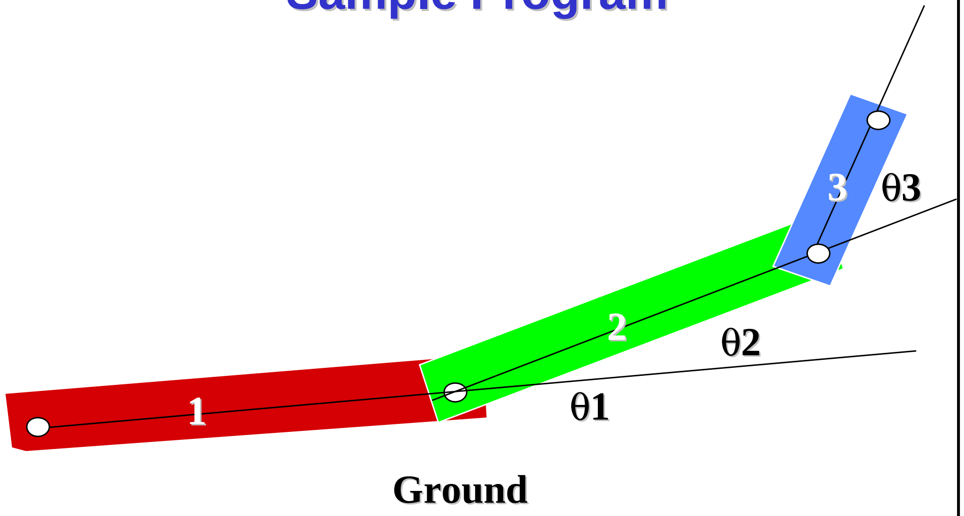


Application Programming Interfaces (APIs)

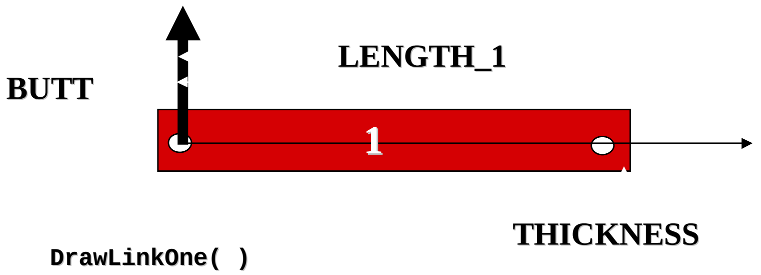
The way the application gains access to the graphics

- OpenGL
- Direct3D
- VRML
- Java 3D
- Fahrenheit

Sample Program



Sample Program



```
DrawLinkOne( )
{
    glBegin( GL_LINE_LOOP );
        glVertex2f(     -BUTT, -THICKNESS/2 );
        glVertex2f( LENGTH_1, -THICKNESS/2 );
        glVertex2f( LENGTH_1,  THICKNESS/2 );
        glVertex2f(     -BUTT,  THICKNESS/2 );
    glEnd();
}
```

Sample Program

```
DrawMechanism( θ1, θ2, θ3 )
    float θ1, θ2, θ3;
{
    glPushMatrix();
        glRotatef( θ1,  0., 0., 1. );
        glIndexi( RED );
        DrawLinkOne();

        glTranslatef( LENGTH_1, 0., 0. );
        glRotatef( θ2,  0., 0., 1. );
        glIndexi( GREEN );
        DrawLinkTwo();

        glTranslatef( LENGTH_2, 0., 0. );
        glRotatef( θ3,  0., 0., 1. );
        glIndexi( BLUE );
        DrawLinkThree();
    glPopMatrix();
}
```

Sample Program

Where in the window to display (pixels)

Viewing Info: field of view angle, x:y aspect ratio, near, far

Whatever interaction is being used

Eye position

```
glViewport( 100, 100,   500, 500 );

glMatrixMode( GL_PROJECTION );
glLoadIdentity( );
gluPerspective( 90., 1.0, 1.,  10. );

glMatrixMode( GL_MODELVIEW );
glLoadIdentity( );

done = FALSE;
while( ! done )
{
    <<  Determine θ1, θ2, θ3  >>
    glPushMatrix();
    gluLookAt( eyex, eyey, eyez,
               centerx, centery, centerz,
               upx, upy, upz );
    DrawMechanism( θ1, θ2, θ3 );
    glPopMatrix();
}
```

Good Geometry References

Foley, Dam, Feiner, Hughes, and Phillips, *Introduction to Computer Graphics*, Addison-Wesley, 1993.

Rogers and Adams, *Mathematical Elements for Computer Graphics*, McGraw-Hill, 1989.

Taylor, *The Geometry of Computer Graphics*, Wadsworth & Brooks/Cole, 1992.

Farin, *Curves and Surfaces for Computer Aided Geometric Design*, Academic Press, 1990.

Good Geometry References

Bartels, Beatty, Barsky, *An Introduction to Splines for use in Computer Graphics and Geometric Modeling*, Morgan-Kaufmann, 1987.

Hoffman, *Geometric & Solid Modeling*, Morgan Kaufmann, 1989.

Mortenson, *Geometric Modeling*, John Wiley & Sons, 1985.

Faux and Pratt, *Computational Geometry for Design and Manufacture*, Ellis-Horwood, 1979.

Graphics Gems 1-5 , Academic Press.